August 1995

UILU-ENG-95-2226

Structures and Algorithms for Two-Dimensional Adaptive Signal Processing

Jeffrey Charles Strait

Coordinated Science Laboratory
College of Engineering
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-95-2226	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) 1308 W. Main St. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

Structures and Algorithms for Two-Dimensional Adaptive Signal Processing

12. PERSONAL AUTHOR(S)
Jeffrey Charles Strait

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM _____ TO _____		140

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	two-dimensional adaptive filters, adoptive filter, two-dimensional system, two-dimensional filtering

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The focus of this work is to explore structures and algorithms for two-dimensional adaptive signal processing. Applications in image and multichannel signal processing include 2-D adaptive differential pulse code modulation, interference cancellation, predictive coding, and noise suppression. Emphasis is placed both on FIR and IIR structures with primary benchmark issues being speed of convergence, computational complexity, and structural flexibility.

(OVER)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

Abstract Cont.

The behavior of the 2-D, FIR, direct form adaptive filter is analogous to that of its 1-D counterpart. Eigenvalue disparity of the input autocorrelation matrix hinders the performance of the steepest descent adaptive algorithm. By implementing a Gauss-Newton sequential adaptive algorithm, the adaptive "modes" are effectively orthogonalized and normalized, thereby increasing the speed of convergence. An efficient block Levinson algorithm is utilized to implement the required matrix operations giving a fast quasi-Newton algorithm (FQN) with $O(N^3)$ complexity. The method exploits the Toeplitz-block Toeplitz structure of the resulting autocorrelation matrix estimate and realizes further computational savings by assuming that the autocorrelation matrix is constant over blocks of N^2 iterations. The FQN filter is compared to the 2-D transform domain filter, the McClellan transformation filter, and the 2-D recursive least squares filter.

Two-dimensional infinite impulse response adaptive filters are also examined. It is found that 2-D IIR adaptive filters are plausible and useful. They exhibit convergence behavior which is dependent upon the 2-D indexing scheme. Several useful indexing methods are examined. A quasi-Newton acceleration algorithm is developed for this structure using the same method as above, except that some additional constraints must be imposed on the 2-D IIR autocorrelation matrix. The 2-D IIR error surface is not quadratic, and must be examined for the possible existence of local minima. Some preliminary results are presented. However, error surfaces can be graphically examined in the three-dimensional coefficient space for IIR filters with first-order denominators. Finally some applications are presented which utilize 2-D IIR adaptive filters. These include 2-D ADPCM and interference cancellation.

STRUCTURES AND ALGORITHMS FOR TWO-DIMENSIONAL ADAPTIVE SIGNAL PROCESSING

BY

JEFFREY CHARLES STRAIT

B.S., University of Illinois, 1986
M.S., University of Illinois, 1989

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1995

Urbana, Illinois

ACKNOWLEDGMENTS

I would like to express grateful acknowledgment to Professor W. Kenneth Jenkins for his patience, support, and technical guidance, and for initiating my study in this area. I would also like to thank my committee members, Professors Douglas Jones, Thomas Huang, and Steven Franke for their helpful suggestions. Also, I thank my wife, Rochelle, and my family for support and encouragement during the completion of this work.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction

Adaptive signal processing has grown at a rapid rate since its introduction in the form of the Least Mean Squares algorithm by Widrow and Hoff in 1959 [1],[2]. Adaptive filtering solutions have been applied to a large number of signal processing problems, primarily in the field of communications [3]-[5]. More than thirty years of intense research on adaptive structures and algorithms has produced a wealth of literature and textbooks on this topic [6]-[12]. Adaptive digital filters are designed for applications in which some parameters of the problem are not known a priori, or they may be changing. Under such circumstances, it is impossible to design a filter which is optimal, and degradations in performance result. Adaptive filters change their filter parameters iteratively in order to strive towards some predefined optimality condition. The ultimate goal of adaptive signal processing is to achieve the best filter performance, usually defined as rate of convergence, with the least amount of computational cost.

Two-dimensional adaptive filtering has been an active area of research recently [13]-[19]. The extension of adaptive filtering techniques to two-dimensional applications such as adaptive differential pulse code modulation, wide band noise suppression, interference cancellation, image restoration and filter design requires adaptive filtering structures and algorithms with reduced computational complexity and fast convergence [6],[20]. These issues are of vital concern because of the large throughput rate required for real time image and video signal processing [17],[21]-[26]. Two-dimensional adaptive filters are recommended because of their ability to take into account the inherent nonstationary statistical properties of images, as well as the two-dimensional statistical correlation present. The classical Wiener filter is not suitable in this environment because

its low-pass characteristics result in blurred edges. Many applications require a linear prediction formulation and are ideal for adaptive filters.

Until recently, these solutions have been focused on finite impulse response (FIR) filter structures such as the direct form structure and the McClellan transformation structure (MCT) [14]-[16],[27]-[31], coupled with traditional steepest descent or least squares adaptive algorithms. The usefulness of 2-D FIR adaptive filters in some of the applications mentioned above has been clearly demonstrated [17],[18],[32]. With the added anticipation of problems requiring filters with inverse modeling capability, two-dimensional infinite impulse response (IIR) filters are proposed as a means of achieving a reduced adaptive parameter set and structural flexibility [33].

One-dimensional (1-D) IIR adaptive filtering has been studied extensively. IIR filters offer better performance than FIR filters because of their ability to effectively model a particular filter response with fewer parameters [34]-[38]. Different IIR filter structures have different performance properties (word length effects, stability, computational complexity) and can be utilized accordingly. Several different 1-D IIR adaptive algorithms have been developed. Those include the Stearns' recursive gradient algorithm [1], the simple hyperstable adaptive recursive filter (SHARF) algorithm [1], the least mean squares equation error (LMSEE) algorithm [35], and the Fan algorithm [39].

Unfortunately, 1-D adaptive recursive filters have several of significant drawbacks. First, it is known that IIR adaptive filters have a nonquadratic mean-square error performance surface which may have local minima, causing the adaptive algorithm to be "stuck" at a suboptimal solution. The 1-D error surface analysis available at this time is from Stearns [40], Soderstrom [41],[42], and Nayeri et al. [43],[44]. Two-dimensional error surface analysis will be conducted using the same fundamental approach, but it will in fact be complicated by the fact that 2-D polynomials can not be factored as in the 1-D case [26]. Second, it is possible for the adaptive process to drive the poles of the IIR filter outside the unit circle, causing instability [35]. This usually requires some form of stability monitoring, which is also difficult for 2-D IIR filters. Infinite impulse response adaptive filtering is generally characterized by slow convergence, which

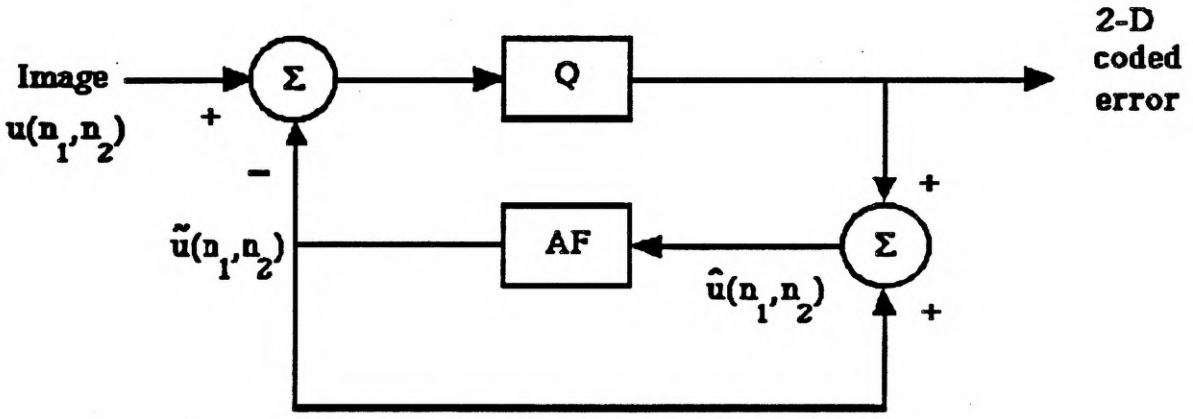
will receive much deserved attention for the proposed 2-D filters. The classical steepest descent algorithm can be modified into a Gauss-Newton algorithm by including second-order input statistics [1]. Fast algorithms are proposed which exploit the block-Toeplitz structure of the 2-D autocorrelation matrix [45].

1.2 Applications of Two-Dimensional Adaptive Filtering

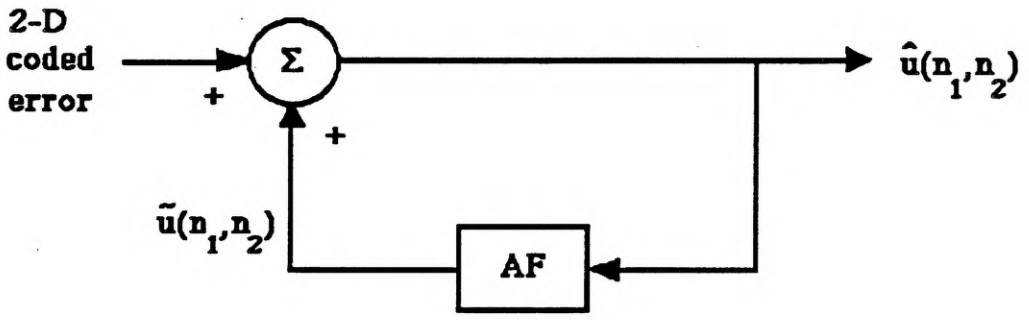
As indicated above, two-dimensional adaptive filtering has as many potential applications as one-dimensional adaptive filtering. Figure 1.1 shows the system configuration for 2-D adaptive differential pulse code modulation. Telephony standards such as CCITT utilize 1-D ADPCM for speech compression over communications links. The CCITT standard ADPCM system uses a four-bit quantizer to achieve a data rate of 32 kbps giving a two-to-one compression ratio. The predictor in this case is an adaptive filter which can account for the input signal's unknown and slowly changing statistics. As a matter of fact, the CCITT ADPCM system uses a pole-zero adaptive filter [46]. Since the error signal, obtained by subtracting the predicted signal from the input signal, is expected to have a much smaller variance than the original signal, it can be coded with fewer bits. Note that the transmitter contains an embedded replica of the receiver, allowing the predictor in the receiver to exactly duplicate the convergence history of the filter in the transmitter. In Figure 1.1 the input signal is an image, and the error signal can be coded and transmitted on-line. For eight-bit gray level images, experimental results have shown that the reconstructed image using a two-bit error word length and a fixed quantizer, Q, is visually indistinguishable from the original image.

Two-dimensional adaptive noise cancellation is shown in Figure 1.2. Just as in the 1-D case, an additive noise signal, $N(n_1,n_2)$, corrupts some desired signal, $s(n_1,n_2)$. The objective is to obtain an estimate of the noise signal and subtract it from the degraded signal to recover an estimate of the original signal, $s(n_1,n_2)$. This technique requires that the adaptive filter "train" on another noise signal which is correlated in some way to the original noise source. The correlation

2-D ADPCM



a) ADPCM Encoder



b) ADPCM Decoder

Standard telephony adpcm : Four-bit quantizer for 32 kbps

Gray level images (8-bit) : Two-bit quantizer gives good visual results using 2-D FIR-LMS

Figure 1.1 Two-dimensional ADPCM a) encoder and b) decoder with quantizer Q.

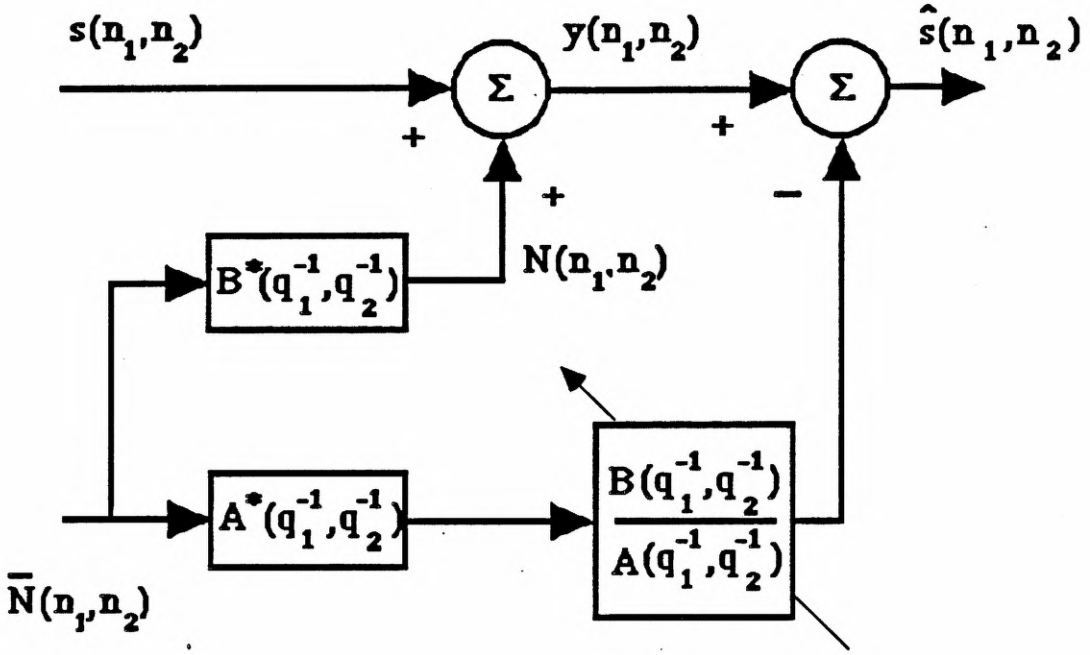


Figure 1.2 Adaptive noise cancellation.

filters between the noise source and the adaptive filter input and measured output signal, $y(n_1,n_2)$, are shown as A^* and B^*, respectively.

Shapiro [17] used a 2-D McClellan transformation filter to adaptively cancel luminance-chrominance crosstalk in frequency domain multiplexed video signals. The video chrominance signal when modulated on the color subcarrier overlaps some of the high frequency luminance information, so that when the composite video signal is decoded degradations are apparent. The problem then is one of noise cancellation similar to that in Figure 1.2. Shapiro designed his enhancement system so that the adaptation was done at the transmitter with filter coefficients being transmitted through a sidechannel in a portion of the vacant video spectrum. The final configuration was equivalent to an adaptive spectrum allocation system.

Figure 1.3 shows an example of wide band noise suppression of 2-D signals. In order to produce an estimate of the desired signal, a narrow band signal degraded with wide band noise is

processed consecutively through a decorrelator delay, $f(q_1^{-1},q_2^{-1})$, and then an adaptive predictor. Based on the same principle as an adaptive line enhancer, the 2-D adaptive filter is unable to predict the noise component of the composite signal so that the output closely approximates the original "clean" image. This technique works very well when low-order, rapidly converging, 2-D adaptive filters are used. The optimal Wiener filter obviously changes constantly as the image statistics change, so that filtering with a fixed low-pass filter will blur sharp edges, resulting in unacceptable visual quality. Many imaging systems, such as low-light vision equipment, scanning electron microscopes, and others, produce images corrupted with wide band noise that are well-suited to this type of enhancement technique.

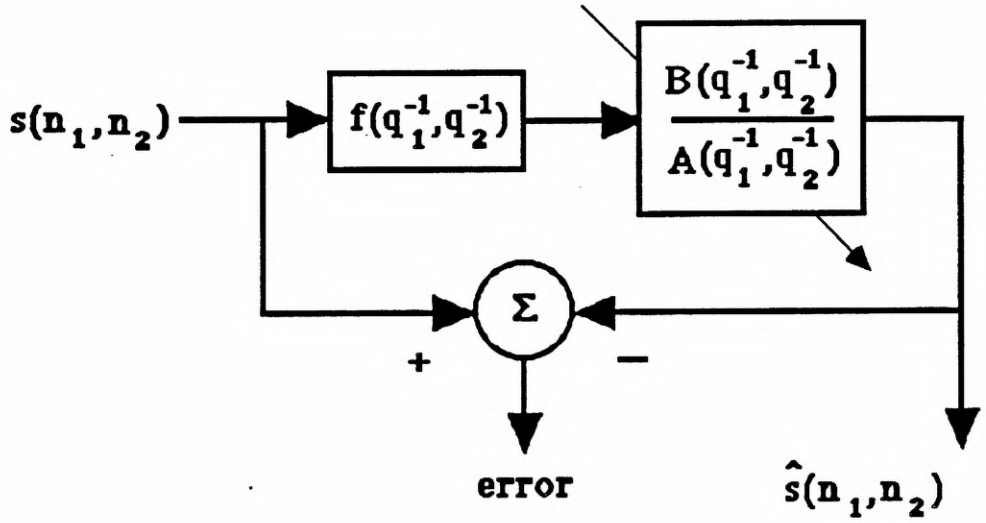


Figure 1.3 Wide band noise suppression.

Another application in which 2-D adaptive filters are useful is image restoration. When some signal is degraded with an unknown system response, the original signal can be recovered with blind equalization. For example, an image may have been blurred with some low-pass point-spread function that can be modeled as an FIR or IIR filter. Adaptive filters can then be configured so that they reproduce the undistorted signal.

In Chapter 2 we explore structures and algorithms for 2-D FIR adaptive filtering. We begin in Section 2.1 with the basic 2-D direct form FIR filter with the LMS algorithm. The Wiener solution for this case is shown to be a direct extension of the 1-D case. More sophisticated methods are examined in Sections 2.2 and 2.3, where the quasi-Newton algorithm is discussed. By incorporating second-order statistics in the steepest descent algorithm, we eliminate the influence of input signal statistics on the rate of convergence. We expect this to significantly improve the performance of the 2-D gradient algorithm for applications of interest. A fast implementation based on the block Levinson algorithm reduces the computational requirements from $O[N^6]$ for brute force inversion to $O[N^3]$. This is in comparison to the $O[N^2]$ requirements of the simple 2-D LMS algorithm. The problem of estimating 2-D autocorrelation lag estimates is given much attention. The performance of the 2-D fast quasi-Newton algorithm relies on the quality of the autocorrelation matrix estimate. In Section 2.4, we present experimental evidence showing the performance improvements.

In Section 2.5, we review the basic principles of the 2-D transform domain adaptive filter. We also give experimental results showing the obvious superiority of the 2-D fast quasi-Newton algorithm over the 2-D transform domain filter. The discrete Fourier transform is chosen for the comparison. In Section 2.6, we present the least squares and recursive least squares solutions for the same 2-D direct form FIR structure. Finally, in Section 2.7, we review the McClellan transformation structure and present some analysis to demystify its remarkable performance.

In Chapter 3 the 2-D IIR structure is covered. The gradient-based adaptive algorithm is derived, and it is shown how, with suitable assumptions, a fast quasi-Newton algorithm can be used as in the FIR case. Experimental results for the 2-D IIR LMS filter and the fast quasi-Newton filter are shown in Sections 3.3 and 3.4. In Section 3.5, uniqueness characteristics of the 2-D IIR mean-squared error minimization are examined. Several simple examples are graphically presented.

Chapter 4 covers applications emphasizing the IIR structure. This is limited to two-dimensional ADPCM and noise cancellation . Finally, conclusions and ideas for further research are discussed at the end of the dissertation.

Standard notation throughout this work is summarized as follows. Two-dimensional spatial signals are always indexed with (n_1,n_2), and the corresponding frequency domain variables with (z_1,z_2) or (ω_1,ω_2). Scalar quantities are denoted by lower case italic characters, such as u, y, and μ. Bold, italicized, lower case characters are vectors, i.e., $\boldsymbol{u}=[u_0 \ldots u_N]^T$. Matrices are represented by bold, italicized capital letters, such as $\boldsymbol{R}$. An "*" superscript indicates either a complex conjugate quantity or an optimal solution. The context in which it is used will be clear.

CHAPTER 2
TWO-DIMENSIONAL FIR ADAPTIVE FILTERING

2.1 The Direct Form Structure with the LMS Algorithm

One-dimensional FIR adaptive filters are widely used because of their simplicity. With a steepest-descent adaptive algorithm such as LMS they offer guaranteed stability and quadratic mean-squared error surfaces [47]-[54]. Structural flexibility is adequate, and in most common circumstances, the computational requirements are low enough so that very long filters can be used if necessary. Some communications applications require filters with hundreds or even thousands of coefficients. For the same reasons, 2-D FIR adaptive filters have been proposed and studied [13]-[17]. Using a simple direct form 2-D FIR structure, many of the familiar adaptive algorithms, such as LMS and RLS, can be extended to 2-D adaptive signal processing. All of the advantages and limitations of 1-D FIR LMS filters extend directly to the 2-D case, and the mathematical analysis of FIR adaptive filters is well-understood. Also, many of the techniques to be utilized in the development of 2-D IIR adaptive filtering are common to 2-D FIR adaptive filtering. For example, later we will design acceleration algorithms based on the Gauss-Newton algorithm that incorporate an autocorrelation matrix estimate into the coefficient update recursion. By utilizing estimates of the input signal's second-order statistics, we can reduce the effects of the input coloring on the adaptive process. This is especially important for 2-D filtering since images contain such dramatic edges. The IIR case will necessarily be approximated so that FIR techniques become embedded in the solution method.

The 2-D FIR filter output is the convolution sum of a simple 2-D tapped delay line and the input

$$y(n_1,n_2) = \sum_{m_1=0}^{N} \sum_{m_2=0}^{N} u(n_1-m_1,n_2-m_2)h_k(m_1,m_2) \tag{2.1}$$

$$= \boldsymbol{h}_k^{\mathrm{T}} \boldsymbol{u}(n_1,n_2) \tag{2.2}$$

where $\boldsymbol{h}_k$, the coefficient vector, and $\boldsymbol{u}(n_1,n_2)$, the information vector, are $(N+1)^2$x1 column-ordered vectors at time-mapped index point k (defined by a suitable indexing method) defined conveniently as

$$\boldsymbol{u}(n_1,n_2) = \begin{bmatrix} u(n_1,n_2) \\ \vdots \\ u(n_1,n_2-N) \\ \vdots \\ u(n_1-N,n_2) \\ \vdots \\ u(n_1-N,n_2-N) \end{bmatrix} \tag{2.3}$$

$$\boldsymbol{h}_k(m_1,m_2) = \begin{bmatrix} h_k(0,0) \\ \vdots \\ h_k(0,N) \\ \vdots \\ h_k(N,0) \\ \vdots \\ h_k(N,N) \end{bmatrix} \tag{2.4}$$

The filter structure above has lower left quarter-plane causality, with the impulse response $\boldsymbol{h}_k(m_1,m_2)$ having a square $(N+1)$x$(N+1)$ region of support. The filter response may be constrained to be linear phase, although here that is not necessarily the case. Equation (2.2) is mathematically equivalent to a length $(N+1)^2$ one-dimensional convolution. The filter coefficients are to be adapted such that an error criterion based on the difference between some desired signal $d(n_1,n_2)$ and the output $y(n_1,n_2)$ is minimized in a system identification configuration. The most common error measure is the mean square error, $\mathrm{E}\{[d(n_1,n_2)-y(n_1,n_2)]^2\}=\mathrm{E}\{e^2(n_1,n_2)\}$.

Assuming a system identification configuration, the optimal Wiener solution can be obtained by setting the gradient of the MSE equal to zero

$$2(\boldsymbol{R}_{uu}\boldsymbol{h}^*(m_1,m_2) - \boldsymbol{p}) = 0 \tag{2.5}$$

giving the Wiener-Hopf solution in terms of the cross-correlation vector and the autocorrelation matrix as

$$\boldsymbol{h}^*(m_1,m_2) = \boldsymbol{R}_{uu}^{-1}\,\boldsymbol{p} \tag{2.6}$$

From above (2.6), the input autocorrelation matrix $\boldsymbol{R}_{uu}$ is defined as

$$\boldsymbol{R}_{uu} = \mathrm{E}\,\{\boldsymbol{u}(n_1,n_2)\,\boldsymbol{u}^{\mathrm{T}}(n_1,n_2)\} = \begin{bmatrix} \boldsymbol{R}_0 & \boldsymbol{R}_{-1} & \cdots & \boldsymbol{R}_{-N} \\ \vdots & \ddots & & \vdots \\ \boldsymbol{R}_N & & \cdots & \boldsymbol{R}_0 \end{bmatrix} \tag{2.7}$$

with each block element being a Toeplitz matrix so that $\boldsymbol{R}_{uu}$ is Toeplitz-block Toeplitz for stationary inputs (Toeplitz-block Toeplitz, or doubly block Toeplitz simply refers to a matrix with both Toeplitz-by-block and block Toeplitz structure combined) [55]

$$\boldsymbol{R}_m = \begin{bmatrix} r_u(m,0) & & \cdots & r_u(m,-N) \\ \vdots & r_u(m,0) & & \vdots \\ & & \ddots & \\ r_u(m,N) & & \cdots & r_u(m,0) \end{bmatrix} \tag{2.8}$$

The autocorrelation lags are defined as

$$r_u(l_1,l_2) = \mathrm{E}\,\{\,u(n_1,n_2)u(n_1-l_1,n_2-l_2)\,\} \tag{2.9}$$

Furthermore, if the input array is separable in addition to being stationary, then (2.8) is symmetric, but that is usually not the case. The cross-correlation vector, $\boldsymbol{p}$, is similarly defined as the expectation of the product of the desired signal and the information vector

$$\boldsymbol{p} = \mathrm{E}\{\, d(n_1,n_2)\boldsymbol{u}(n_1,n_2)\,\} \tag{2.10}$$

A 2-D LMS coefficient update may be obtained in a straightforward manner using the steepest descent algorithm (2.11) with Widrow's LMS gradient approximation (2.12) [13]

$$\boldsymbol{h}_{k+1} = \boldsymbol{h}_k - \mu\hat{\nabla}_h(\xi) \tag{2.11}$$

where

$$\xi = \mathrm{E}\{e^2(n_1,n_2)\} \approx e^2(n_1,n_2) \tag{2.12}$$

so that

$$h_{k+1}(m_1,m_2) = h_k(m_1,m_2) + 2\mu e(n_1,n_2)u(n_1\text{-}m_1,n_2\text{-}m_2) \tag{2.13}$$

This can be written collectively as

$$\boldsymbol{h}_{k+1} = \boldsymbol{h}_k + 2\,\mu e(n_1,n_2)\boldsymbol{u}(n_1,n_2) \tag{2.14}$$

using the column-ordered vector representation of $\boldsymbol{h}_k$ and $\boldsymbol{u}(n_1,n_2)$. Again, k is some function of (n_1,n_2) specifying the indexing scheme. The behavior of the 2-D LMS algorithm (2.14) is well-understood. The performance analysis of (2.14) is identical to that of the 1-D LMS algorithm using the appropriate 2-D statistical quantities. Just as in the 1-D case, when the input process is colored so that the 2-D autocorrelation matrix (2.7) has disparate eigenvalues, the rate of

convergence is severely reduced. The error surface remains quadratic in this case, but it becomes elliptical, rotated, translated and stretched out so that the steepest descent search direction is no longer a direct path to the optimal solution given in (2.6). The time constant for each mode in the adaptive process is inversely proportional to each corresponding eigenvalue of the autocorrelation matrix. But now, the adaptive parameter set has $(N+1)^2$ coefficients, which hampers the rate of convergence, since we know that increasing the number of parameters reduces the rate of convergence [1]. Also, throughout this work it is assumed that all adaptive filters are used with local mean estimators to eliminate the effects of nonzero mean signal values on the adaptive process.

2.2 The Quasi-Newton Algorithm

To compensate for the performance penalty of using the LMS algorithm in the presence of colored input signals, we will modify the algorithm from a steepest descent search to an approximated Newton algorithm [1],[47],[56]-[58]. The primary objection to the quasi-Newton algorithm in adaptive filtering has always been the high cost in terms of computational complexity. The simple LMS algorithm requires only $O[N]$ complexity for 1-D FIR adaptive filters and $O[N^2]$ for 2-D FIR adaptive filters. Quasi-Newton algorithms require inversion of the autocorrelation matrix, or, equivalently, the $(N+1)^2 \times (N+1)^2$ system solution of $\boldsymbol{R}_{uu}\boldsymbol{x} = \boldsymbol{b}$. Solving the system by brute force with Gaussian elimination requires $O[N^3]$ for 1-D problems and $O[N^6]$ for 2-D problems. This order of complexity is outrageous when considering real-time signal processing applications. Later, we will show fast implementation methods that reduce the required complexity to a more reasonable level. The quasi-Newton algorithm provides convergence acceleration with better numerical properties and better tracking performance than the recursive least squares method. The MSE performance function for 2-D FIR adaptive filters is quadratic and is given as

$$\xi = \mathrm{E}\{d^2(n_1,n_2)\} + \boldsymbol{h}^{\mathrm{T}}(m_1,m_2)\boldsymbol{R}_{uu}\boldsymbol{h}(m_1,m_2) - 2\boldsymbol{p}^{\mathrm{T}}\boldsymbol{h}(m_1,m_2) \tag{2.15}$$

Since (2.15) is quadratic with $\boldsymbol{R}_{uu}$ positive definite, Newton's method can achieve one step convergence to the optimal solution (2.6) as

$$\boldsymbol{h}_{k+1}(m_1,m_2) = \boldsymbol{h}_k(m_1,m_2) - \mu \boldsymbol{R}_{uu}^{-1}\nabla(\xi) \tag{2.16}$$

$$\boldsymbol{h}_{k+1}(m_1,m_2) = \boldsymbol{h}_k(m_1,m_2) - \mu\boldsymbol{R}_{uu}^{-1}(2\boldsymbol{R}_{uu}\boldsymbol{h}_k(m_1,m_2) - 2\,\boldsymbol{p}) \tag{2.17}$$

$$= \boldsymbol{h}_k(m_1,m_2) - 2\mu\,\mathbf{I}\,\boldsymbol{h}_k(m_1,m_2) + 2\mu\boldsymbol{R}_{uu}^{-1}\boldsymbol{p} \tag{2.18}$$

$$= (1-2\mu)\boldsymbol{h}_k(m_1,m_2) + 2\mu\boldsymbol{h}^*(m_1,m_2) \tag{2.19}$$

with μ=1/2

$$\boldsymbol{h}_{k+1}(m_1,m_2) = \boldsymbol{h}^*(m_1,m_2) \tag{2.20}$$

However, (2.16) can not be implemented in exact form in practice for several reasons. First of all, the gradient vector, $\nabla(\xi)$, is not generally known and must be estimated. The same is true with the autocorrelation matrix. However, it is possible to estimate both of these quantities on-line and include them in an iterative quasi-Newton (QN) algorithm as follows

$$\boldsymbol{h}_{k+1}(m_1,m_2) = \boldsymbol{h}_k(m_1,m_2) - \frac{1}{2}\mu\,\hat{\boldsymbol{R}}_{uu}^{-1}\hat{\nabla}_k \tag{2.21}$$

The LMS gradient vector approximation can be combined with the QN algorithm so that (2.21) becomes

$$\boldsymbol{h}_{k+1}(m_1,m_2) = \boldsymbol{h}_k(m_1,m_2) + \mu e(n_1,n_2)\hat{\boldsymbol{R}}_{uu}^{-1}\boldsymbol{u}(n_1,n_2) \tag{2.22}$$

Widrow and Stearns [1] analyzed the performance improvement of the QN algorithm over that for the LMS algorithm for the 1-D case. Assuming that the autocorrelation matrix has $(N+1)^2$

eigenvalues $\lambda_0 \leq \lambda_1 \ldots \leq \lambda_{(N+1)^2-1}$, then the learning-curve time constants for the LMS and QN algorithms are

$$\text{LMS} \quad : \qquad T_{\text{mse},n} = \frac{1}{4\mu\lambda_n} \qquad n = 0, \ldots, (N+1)^2-1 \tag{2.23}$$

$$\text{QN} \quad : \qquad T_{\text{mse}} = \frac{1}{4\mu\lambda_{\text{avg}}} \tag{2.24}$$

When the input process is white, $\lambda_{\text{avg}} = \lambda_n$ for $n = 0, \ldots, (N+1)^2-1$. For colored input signals, the longest time constant in the LMS learning curve is always larger than the QN time constant, since $\lambda_{\text{min}} < \lambda_{\text{avg}}$. Therefore, the performance limit of the quasi-Newton algorithm using the LMS gradient vector approximation with a stationary input is simply that of the convergence performance of the standard LMS filter in the presence of white noise. The gradient-based adaptive algorithms are sometimes normalized by including a power estimate in the denominator of the update relation. This becomes very important when using the quasi-Newton algorithm with nonstationary inputs.

2.3 Fast Quasi-Newton Implementation

Consider first the problem of estimating the input autocorrelation matrix (2.7). The quasi-Newton algorithm uses the inverse of $\hat{\boldsymbol{R}}_{uu}$, but $\hat{\boldsymbol{R}}_{uu}^{-1}$ can not be easily estimated since little is known about the properties of the inverse autocorrelation matrix. However, we do know much about the properties of the autocorrelation matrix, so it makes sense to construct an estimate of $\boldsymbol{R}_{uu}$ and use that in a fast inversion routine. For example, we know that for stationary input signals, $\boldsymbol{R}_{uu}$ is Toeplitz-block Toeplitz and Hermitian symmetric. Therefore, the estimation problem then becomes an estimation of each of the $2(N+1)^2-N$ autocorrelation lag coefficients. The Toeplitz-block Toeplitz structure can then be forcibly imposed on the matrix. We know also that $\boldsymbol{R}_{uu}$ is positive semidefinite, as it must be to insure invertibility. In fact, $\boldsymbol{R}_{uu}$ is almost always positive

definite except for cases when the input is not persistently exciting. The positive semidefinite condition can be stated as

$$\boldsymbol{v}^T\boldsymbol{R}_{uu}\boldsymbol{v} = \sum_{s=0}^{(N+1)^2-1} \sum_{t=0}^{(N+1)^2-1} v_s[\boldsymbol{R}_{uu}]_{s,t}v_t \geq 0 \tag{2.25}$$

Equation (2.25) follows from the linearity property of expected values

$$\mathrm{E}\{|v_0u_0 + v_1u_1 + \ldots + v_pu_p|^2\} = \sum_{s=0}^{p} \sum_{t=0}^{p} v_s[\boldsymbol{R}_{uu}]_{s,t}v_t \tag{2.26}$$

where u_i is the i^{th} element of $u(n_1,n_2)$ and $p=(N+1)^2-1$. Therefore, if the autocorrelation lag estimates represent a valid 2-D autocorrelation sequence, then the sequence, as well as the matrix, are positive semidefinite by definition. A simple way to guarantee the positive definite property is to then add $\delta\boldsymbol{I}$ to $\hat{\boldsymbol{R}}_{uu}$, where δ is a small positive constant.

Because of the symmetric Toeplitz-block Toeplitz structure of $\boldsymbol{R}_{uu}$, there are actually only $2(N+1)^2-N$ distinct lag estimates which must be computed. We need to derive an efficient, recursive algorithm to compute each consecutive element from new data and its value at the preceding index points. Recognizing that adaptive filters are often used in nonstationary environments, an exponentially weighted recursion is ideal since that would allow the algorithm to "forget" old data when estimating $\boldsymbol{R}_{uu}$. A suitable starting point is to consider a biased autocorrelation lag coefficient estimator such as

$$\hat{r}_u(k_1,k_2) = \frac{1}{n_1n_2} \sum_{m_1=k_1}^{n_1} \sum_{m_2=k_2}^{n_2} u(m_1-k_1,m_2-k_2)u^*(m_1,m_2) \tag{2.27}$$

The unbiased counterpart to (2.27) has the divisor n_1n_2 replaced by $(n_1-k_1)(n_2-k_2)$. Generally, (2.27) is preferred over the unbiased version because it usually provides a positive semidefinite

autocorrelation estimate over a block of data [59]. The unbiased estimator is more likely to produce an invalid autocorrelation sequence because of the triangular-shaped normalizing factor. Including an exponential weighting factor, $0 \le \alpha \le 1$, (2.27) becomes

$$\hat{r}_{n_1 n_2}(k_1,k_2) = K(n_1,n_2) \sum_{m_1=k_1}^{n_1} \sum_{m_2=k_2}^{n_2} \alpha^{(n_1-m_1+n_2-m_2)} u(m_1-k_1,m_2-k_2) u^*(m_1,m_2) \tag{2.28}$$

$$\begin{aligned} &= K(n_1,n_2)\ \alpha \sum_{m_1=k_1}^{n_1-1} \sum_{m_2=k_2}^{n_2} \alpha^{(n_1-1-m_1+n_2-m_2)} u(m_1-k_1,m_2-k_2) u^*(m_1,m_2) \\ &+ K(n_1,n_2)\ \alpha \sum_{m_1=k_1}^{n_1} \sum_{m_2=k_2}^{n_2-1} \alpha^{(n_1-m_1+n_2-1-m_2)} u(m_1-k_1,m_2-k_2) u^*(m_1,m_2) \\ &- K(n_1,n_2)\ \alpha^2 \sum_{m_1=k_1}^{n_1-1} \sum_{m_2=k_2}^{n_2-1} \alpha^{(n_1-1-m_1+n_2-1-m_2)} u(m_1-k_1,m_2-k_2) u^*(m_1,m_2) \\ &+ K(n_1,n_2)\ u(n_1-k_1,n_2-k_2) u^*(n_1,n_2) \end{aligned} \tag{2.29}$$

$$\begin{aligned} &= \alpha \hat{r}_{n_1-1 n_2}(k_1,k_2)\ \frac{K(n_1,n_2)}{K(n_1-1,n_2)} + \alpha \hat{r}_{n_1 n_2-1}(k_1,k_2)\ \frac{K(n_1,n_2)}{K(n_1,n_2-1)} \\ &\quad - \alpha^2 \hat{r}_{n_1-1 n_2-1}(k_1,k_2)\ \frac{K(n_1,n_2)}{K(n_1-1,n_2-1)} \\ &\quad + K(n_1,n_2) u(n_1-k_1,n_2-k_2) u^*(n_1,n_2) \end{aligned} \tag{2.30}$$

The normalization factor can be chosen as follows

$$\mathrm{E}\{\hat{r}_{n_1 n_2}(k_1,k_2)\} = \mathrm{E}\left\{ K(n_1,n_2) \sum_{m_1=k_1}^{n_1} \sum_{m_2=k_2}^{n_2} \alpha^{(n_1-m_1+n_2-m_2)} u(m_1-k_1,m_2-k_2) u^*(m_1,m_2) \right\}$$

$$= K(n_1,n_2) \sum_{m_1=k_1}^{n_1} \sum_{m_2=k_2}^{n_2} \alpha^{(n_1-m_1+n_2-m_2)}\, \mathrm{E}\{u(m_1-k_1,m_2-k_2) u^*(m_1,m_2)\}$$

$$= K(n_1,n_2)\, r_{n_1n_2}(k_1,k_2) \sum_{m_1=k_1}^{n_1} \sum_{m_2=k_2}^{n_2} \alpha^{(n_1-m_1+n_2-m_2)}$$

$$= K(n_1,n_2)\, r_{n_1n_2}(k_1,k_2) \frac{\alpha^{n_1-k_1+1} - 1}{\alpha-1} \frac{\alpha^{n_2-k_2+1} - 1}{\alpha-1} \tag{2.31}$$

so that arbitrarily initiating the indexing point (m_1,m_2) at (1,1) we have

$$K(n_1,n_2) = \frac{(1-\alpha)^2}{(1-\alpha^{n_1})(1-\alpha^{n_2})} \tag{2.32}$$

Equations (2.30) and (2.32) combine to give a reasonable autocorrelation lag estimate recursion formula for 2-D data.

However, if we modify the estimate so that the index block length in the summation in (2.29) is fixed, we must exponentially weight the data before computing our lag estimate in order for it to remain biased. Equation (2.29) can equivalently be considered to be averaged data, with the appropriate spatial lags, scaled by $(\alpha^{n_1-m_1})^{1/2}(\alpha^{n_2-m_2})^{1/2}$. If we modify the scale factor to become $(\alpha^{n_1-m_1+k_1})^{1/2}(\alpha^{n_2-m_2+k_2})^{1/2}$, then each lag estimate will be biased by the factor $(\alpha^{k_1})^{1/2}(\alpha^{k_2})^{1/2}$, so that as the spatial lag increases update terms become increasingly biased. The value of α is usually chosen to be near to 1, but this choice depends on the statistical properties of the input process. For large values of n_1 and n_2, the denominator of (2.32) approaches 1. In one-dimensional applications, the exponential terms are usually negligible and can be assumed to be one. But, in 2-D signal processing the ratio of these terms can not easily be ignored since they do affect the integrity of the estimate. Therefore, we now have a suitable, low-complexity estimate that can be included in (2.22).

$$\hat{r}_{n_1n_2}(k_1,k_2) = \alpha\, \hat{r}_{n_1-1n_2}(k_1,k_2) \frac{(1-\alpha^{n_1-1})}{(1-\alpha^{n_1})} + \alpha\, \hat{r}_{n_1n_2-1}(k_1,k_2) \frac{(1-\alpha^{n_2-1})}{(1-\alpha^{n_2})}$$

$$- \alpha^2 \hat{r}_{n_1-1 n_2-1}(k_1,k_2) \frac{(1-\alpha^{n_1-1})(1-\alpha^{n_2-1})}{(1-\alpha^{n_1})(1-\alpha^{n_2})}$$

$$+ \frac{(1-\alpha)^2}{(1-\alpha^{n_1})(1-\alpha^{n_2})} u(n_1-k_1,n_2-k_2)u^*(n_1,n_2) \tag{2.33}$$

In order to use the acceleration algorithm shown in (2.22), a fast algorithm to solve the Toeplitz-block Toeplitz system of equations, $\boldsymbol{R}_{uu}\boldsymbol{x}=\boldsymbol{u}(n_1,n_2)$, must be utilized. Solving for $\boldsymbol{x}$ by brute force at each iteration is simply too expensive to be considered for a sequential adaptive algorithm.

The block Levinson algorithm, however, is suitable for use in a 2-D sequential adaptive update recursion with reduced computational complexity [20],[45],[55]. The block Levinson algorithm algebraically solves a block Toeplitz system of equations with O[N^5] complexity. The proposed implementation requires a block formulation with the autocorrelation matrix assumed to be constant over a block of data. Many 2-D signals and images are nonstationary in general, but they often have large regions which may be assumed to be stationary. An image can be preprocessed with one of the many image segmentation algorithms available to identify blocks that are approximately stationary and, hence, have a constant autocorrelation matrix.

Marshall demonstrated the utility of the Levinson algorithm in implementing a fast 1-D block quasi-Newton algorithm [58]. The Levinson algorithm solves a positive definite Toeplitz system of equations in O[N^2] complexity, which does not offer any computational improvement for the quasi-Newton algorithm over the recursive least squares algorithm. However, Marshall noticed that if the input signal is assumed to be stationary over finite length blocks as above, then it is possible to compute successive solutions of $\boldsymbol{Rx}=\boldsymbol{b}$ with only O[N] computations per iteration within each block following the initial Levinson solution. This relies on the fact that the input vector, $\boldsymbol{b}$, has the sliding property. That is, successive input vectors are shifted versions of each other with the current input sample shifted into the leading element of $\boldsymbol{b}$. The procedure, outlined above, of computing consecutive solutions of $\boldsymbol{Rx}=\boldsymbol{b}$ is exactly analogous to determining optimum

Wiener filters for consecutive time lags [45]. Block algorithms often offer computational and performance advantages, and they have been developed by several different authors [60]-[66].

If the 1-D block quasi-Newton algorithm processes blocks of length N or greater, then the overall complexity is $O[(N^2+N(N-1))/N] = O[N]$. This corresponds to a length $2N$ block of input samples for a length N filter computing N output samples. It seems to be reasonable to assume that length $2N$ blocks of data are stationary in many instances, and experimentation has confirmed that the extent of degradation resulting from this assumption is minimal.

Similarly, the block Levinson algorithm can be used to solve for the product of the inverse autocorrelation matrix and the gradient vector in a 2-D block quasi-Newton adaptive algorithm. Algebraic manipulation of the block Levinson algorithm yields an efficient method of computing successive solutions of the Toeplitz-block Toeplitz system $\boldsymbol{R}_{uu}\boldsymbol{x}=\boldsymbol{u}(n_1,n_2)$, just as was the case for 1-D Toeplitz systems. This gives us a fast way to compute the next solution, $\boldsymbol{x}_{k+1}=\hat{\boldsymbol{R}}_{uu}^{-1}\boldsymbol{u}(n_1,n_2)$, from $\boldsymbol{x}_k=\hat{\boldsymbol{R}}_{uu}^{-1}\boldsymbol{u}(n_1-1,n_2)$. The input vector $\boldsymbol{u}(n_1,n_2)$ is now a column (or row) ordered $(N\text{x}1)^2\text{x}1$ vector constructed by concatenating columns (or rows) of the 2-D input signal over the $N\text{x}N$ region of support of the adaptive filter mask. The autocorrelation matrix, $\boldsymbol{R}$, is now $(N+1)^2\text{x}(N+1)^2$, and consecutive input vectors are vector-shifted versions of one another. That is, length N element vectors are shifted through $\boldsymbol{u}(n_1,n_2)$ as the adaptive filter indexes through the input image, i.e.,

$$\boldsymbol{u}(n_1-1,n_2) = \begin{bmatrix} u(n_1-1,n_2) \\ \vdots \\ u(n_1-1,n_2-N) \\ u(n_1-2,n_2) \\ \vdots \\ u(n_1-N,n_2-N) \end{bmatrix} \tag{2.34}$$

$$u(n_1,n_2) = \begin{bmatrix} u(n_1,n_2) \\ \vdots \\ u(n_1,n_2\text{-}N) \\ u(n_1\text{-}1,n_2) \\ \vdots \\ u(n_1\text{-}N,n_2\text{-}N) \end{bmatrix} \tag{2.35}$$

Now, the block Levinson algorithm requires matrix multiplications whereas the Levinson algorithm required scalar multiplications. This accounts for the fact that the initial block Levinson solution requires $O[N^5]$ complexity, and the successive spatial lag solutions require $O[N^3]$. Therefore, a block algorithm implemented within a block of N^2 pixels of the 2-D output mask requires $\{O[N^5] + N^2O[N^3]\}/N^2 = O[N^3]$ complexity. At this time, we require run length blocks of data to be length N^2 since there appears to be no computationally efficient means of indexing between rows (for a horizontally indexed rectangular grid). Specifically, this means that we must assume the stationary block of data is of size $(N^2+N)xN$. The block Levinson algorithm is listed in Table 2.1 without proof [45]. The recursion algorithm for each of the next N consecutive solutions is given in Table 2.2. The matrix $\boldsymbol{J}$ is a block matrix whose elements on the antidiagonal are identity matrices, and the remaining matrices are zero. The "step-up" recursion in Table 2.2, with the input generically labeled $\boldsymbol{b}$

$$\boldsymbol{x}_p^{l+1} = \begin{bmatrix} \boldsymbol{x}_{p-1}^{l+1} \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_{p-1} \\ 1 \end{bmatrix} k_p^{l+1} \tag{2.36}$$

is generated simply as the last block Levinson recursion in Table 2.1. To show the "step-down" recursion, we make use of the persymmetry property of block Toeplitz matrices, $\boldsymbol{R}_m^T\boldsymbol{J}=\boldsymbol{J}\boldsymbol{R}_m$, and show the relationship of two succeeding systems.

$$\boldsymbol{R}_p\boldsymbol{x}_p^l = \begin{bmatrix} \boldsymbol{b}_l \\ \boldsymbol{b}_{l+1\ l+p-1} \end{bmatrix} \tag{2.37}$$

Table 2.1 The block Levinson algorithm for solving Toeplitz-block Toeplitz systems of equations.

solve: $\boldsymbol{R}_p \boldsymbol{x}_p^l = \boldsymbol{b}_{l,l+p-1}$

$$
\begin{aligned}
\boldsymbol{x}_{m+1}^{l} &= \begin{bmatrix} \boldsymbol{x}_m^l \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_m \\ 1 \end{bmatrix} \boldsymbol{k}_{m+1}^{l} \\
\boldsymbol{J}\boldsymbol{w}_{m+1} &= \begin{bmatrix} \boldsymbol{J}\boldsymbol{w}_m \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{f}_m \\ 1 \end{bmatrix} \boldsymbol{k}_{m+1}^{w} \\
\boldsymbol{J}\boldsymbol{f}_{m+1} &= \begin{bmatrix} \boldsymbol{J}\boldsymbol{f}_m \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_m \\ 1 \end{bmatrix} \boldsymbol{k}_{m+1}^{f} \\
\boldsymbol{k}_{m+1}^{l} &= -\alpha_m^{-1}\beta_m^{l} \\
\boldsymbol{k}_{m+1}^{w} &= -\delta_m^{-1}\beta_m^{w}
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{k}_{m+1}^{f} &= -\alpha_m^{-1}\beta_m^{f} \\
\beta_m^{l} &= \boldsymbol{w}_m^{T}\, \boldsymbol{b}_{l,l+m-1} + \boldsymbol{b}_{l+m} = \boldsymbol{r}_m^{\mathrm{T}}\, \boldsymbol{J}\boldsymbol{x}_m^{l} + \boldsymbol{b}_{l+m} \\
\beta_m^{w} &= \boldsymbol{r}_m^{\mathrm{T}}\, \boldsymbol{w}_m + \boldsymbol{r}_{m+1} \\
\beta_m^{f} &= \boldsymbol{r}_m^{\mathrm{T}}\boldsymbol{f}_m + \boldsymbol{r}_{m+1} \\
\alpha_m &= \alpha_{m-1} + \beta_{m-1}^{f}\, \boldsymbol{k}_m^{w} \\
\delta_m &= \delta_{m-1} + \beta_{m-1}^{w}\, \boldsymbol{k}_m^{f} \\
(\beta_m^{w})^{\mathrm{T}} &= \beta_m^{f} \\
\det \alpha_m &= \det \delta_m \\
\boldsymbol{r}_m &= [\, r_1\, r_2 \ldots r_m]^{\mathrm{T}}
\end{aligned}
$$

Table 2.2 The block Levinson algorithm for efficiently solving the next Toeplitz-block Toeplitz system of equations, given parameters from the original Levinson solution.

$$\begin{bmatrix} Jx_{p-1}^{l+1} \\ 0 \end{bmatrix} = Jx_p^l - \begin{bmatrix} f_{p-1} \\ 1 \end{bmatrix} \hat{k}_{p-1}^l$$

$$x_p^{l+1} = \begin{bmatrix} x_{p-1}^{l+1} \\ 0 \end{bmatrix} + \begin{bmatrix} w_{p-1} \\ 1 \end{bmatrix} k_p^{l+1}$$

$$k_p^{l+1} = -\alpha_{p-1}^{-1} \beta_{p-1}^{l+1}$$

$$\beta_{p-1}^{l+1} = r_{p-1}^T J x_{p-1}^{l+1} + b_{l+p}$$

$$= w_{p-1}^T b_{l+1,l+p-1} + b_{l+p}$$

$$R_p x_p^{l+1} = \begin{bmatrix} b_{l+1\ l+p-1} \\ b_{l+p} \end{bmatrix} \tag{2.38}$$

$$R_p J x_p^l = \begin{bmatrix} Jb_{l+1\ l+p-1} \\ b_l \end{bmatrix} \tag{2.39}$$

Now, from $R_{p-1} x_{p-1}^{l+1} = b_{l+1,l+p-1}$ we obtain

$$R_{p-1} J x_{p-1}^{l+1} = Jb_{l+1\ l+p-1} \tag{2.40}$$

so that the solutions of (2.39) and (2.40) are clearly related through Levinson's algorithm as

$$\begin{bmatrix} Jx_{p-1}^{l+1} \\ 0 \end{bmatrix} = Jx_p^l - \begin{bmatrix} f_{p-1} \\ 1 \end{bmatrix} \hat{k}_{p-1}^l \tag{2.41}$$

A two-dimensional filter indexed vertically or horizontally can be described mathematically as a multichannel filter for the duration of the scan line. Usually multichannel filters have a spatial index and a time index, but there is no reason to distinguish between the two for the purpose of analysis. We just consider the fact that both problems have two dependent variables. Therefore, both of these algorithms are valid for multichannel filter formulations. In fact, the block Levinson algorithm with optimum time-lag filters was specifically designed for multichannel filters. They can be applied to both FIR filters and IIR filters as long as the autocorrelation matrix is Toeplitz-block Toeplitz and the input vectors have the "sliding" property. Somewhat of a concern is the $O[N^3]$ complexity required. While not a drawback for IIR adaptive filters with small parameter sets, FIR filters can have relatively large regions of support and $O[N^3]$ complexity can become burdensome, although it appears that most image processing applications require relatively short filters. By contrast, the 2-D LMS algorithm has $O[N^2]$ complexity, and the 2-D RLS algorithm has $O[N^4]$ (for an NxN filter). This clearly shows the tradeoff between algorithm performance and computational complexity. In light of the numerical problems associated with the RLS algorithm, the 2-D block quasi-Newton method should prove to be a viable and efficient solution.

Nonstationary inputs must be considered since most real signals are nonstationary. The spatial dependence of the input signal's statistics is limitless, but it may be advantageous to consider some simple possibilities. For example, we can consider the case in which the statistics change abruptly only at horizontal and vertical edges. Assuming that we are using horizontal indexing, it seems straightforward to examine the case in which a vertical boundary exists which separates regions of an image with different statistical characteristics. The algorithm can be designed to index all the way to the right-most edge of the image, in which case, the values of the lag coefficients exponentially approach the new values determined by the statistics of that block. The rate at which they converge to the correct values depends on the forgetting factor built into the lag coefficient update recursion. Until that point is reached, the FQN algorithm will be using an eroneous autocorrelation estimate, which will hamper performance. A second approach, hinted at in the design of the lag recursion, is to design the indexing scheme to process short rows

consecutively at the boundary of a vertical edge, i.e., process rectangular blocks while indexing the blocks vertically. That does not resolve the dilemma since we must eventually index over to the next column of blocks at which time the filter will overlap pixels with differing statistical properties. If we then consider processing blocks of pixels as above with horizontal indexing a vertical edge causes the filter to overlap nonstationary data, and the processing for that row will be suboptimal. These issues are the source of the limitations of the 2-D FQN algorithm.

The preconditioned conjugate gradient (PCG) method offers an attractive alternative method for solving a Toeplitz-block Toeplitz system of equations of the form $\boldsymbol{Tx}=\boldsymbol{b}$ [67]. The conjugate gradient algorithm is an iterative solution method derived by minimizing the quadratic functional $0.5(\boldsymbol{x}^t\boldsymbol{T}\boldsymbol{x}-\boldsymbol{b}^t\boldsymbol{x})$ using a different search direction [67]-[72]. In general, the conjugate gradient method will converge faster to a unique solution if the eigenvalues of the matrix $\boldsymbol{T}$ are clustered around 1. Preconditioning is commonly used to improve the clustering property and accelerate convergence. Just as in any steepest descent algorithm, the properties of the error surface govern the rate of convergence of the PCG. The preconditioned conjugate gradient algorithm is listed without proof [67] in Table 2.3. $\boldsymbol{P}$ is the circulant-block circulant preconditioner. For an N^2xN^2 system, the computational burden of the PCG algorithm is incurred while solving $\boldsymbol{P}\boldsymbol{z}_{k-1}=\boldsymbol{r}_{k-1}$ and computing the matrix vector product $\boldsymbol{T}\boldsymbol{p}_k$. All of the vector-vector and vector-scalar products contribute $O[N^2]$ per iteration towards computational complexity. Since $\boldsymbol{T}$ is Toeplitz-block Toeplitz, $\boldsymbol{T}\boldsymbol{p}_k$ can also be computed with 2-D FFTs in $O[N^2\log_2 N]$ multiplications per iteration. This follows since 2-D convolution can be expressed as the product of a Toeplitz-block Toeplitz input matrix and a column-ordered vector. It follows that $\boldsymbol{P}\boldsymbol{z}_{k-1}=\boldsymbol{r}_{k-1}$ can also be solved using the 2-D FFT in $O[N^2\log_2 N]$.

Until recently the block PCG algorithm was not considered adequate for use in a sequential adaptive algorithm because convergence of the PCG algorithm itself was $O[N]$ for most preconditioners. The overall $O[N^3\log_2 N]$ complexity would limit the usefulness in real-time applications. Several excellent preconditioners have been proposed in order to improve BPCG performance. Ku and Kuo [67] proposed a circulant-block circulant preconditioner of the form

Table 2.3 The preconditioned conjugate gradient algorithm.

for arbitrary x_0 $\quad r_0=p_0=b-Tx_0$

$\beta_1=0$

for k=1, 2, ...

$$Pz_{k-1} = r_{k-1}$$

$$\beta_k = (z_{k-1},r_{k-1})/(z_{k-2},r_{k-2})$$

$$p_k = z_{k-1} + \beta_k p_{k-1}$$

$$\alpha_k = (z_{k-1},r_{k-1})/(p_k,Tp_k)$$

$$x_k = x_{k-1} + \alpha_k p_k$$

$$r_k = r_{k-1} - \alpha_k T p_k$$

$$P = \begin{bmatrix} P_0 & P_1 & \cdots & P_{N-1} \\ P_{N-1} & & \cdots & P_{N-2} \\ & & \cdots & \\ P_1 & & \cdots & P_{N-1} \; P_0 \end{bmatrix} \tag{2.42}$$

where P_n, $0 \le n \le N-1$, are NxN circulant matrices with elements defined as

$$P_n = \begin{cases} C_0 + cI & n=0 \\ C_n + C_{N-n} & 1 \le n < N \end{cases} \tag{2.43}$$

$$[C_n]_{ij} = [T_{n,1}]_{ij} + [T_{n,2}]_{ij} \tag{2.44}$$

$$T_{n,1} = \begin{bmatrix} t_0 & t_1 & \cdots & t_{N-1} \\ \cdots & \cdots & & \cdots \\ t_{N-1} & \cdots & & t_0 \end{bmatrix} \tag{2.45}$$

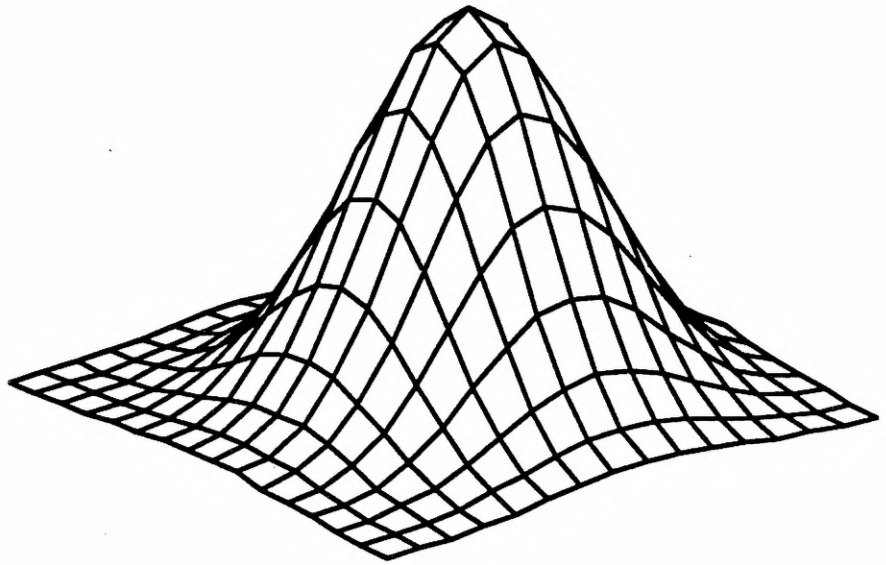
$$T_{n,2} = \begin{bmatrix} c & t_{N-1} & \cdots & t_1 \\ \cdots & \cdots & & \cdots \\ t_1 & \cdots & t_{N-1} & c \end{bmatrix} \tag{2.46}$$

The block PCG with this preconditioner can be shown to converge in a fixed number of iterations for large N. Therefore, the overall computational complexity is $O[N^2 \log_2 N]$. This is a remarkable result which allows the quasi-Newton algorithm to approach the efficiency of the $O[N^2]$ LMS.

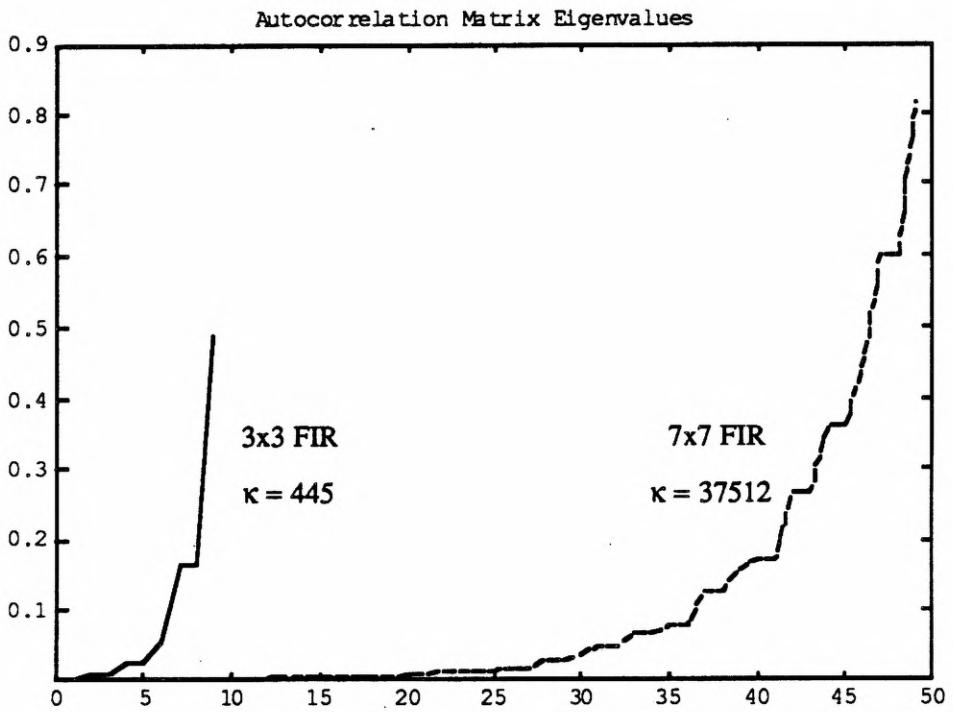
2.4 Experimental Results for the 2-D Fast Quasi-Newton Algorithm

From the last section, we recognize that if the input to an adaptive filter is colored, then the eigenvalues of the autocorrelation matrix are disparate. Only in the white noise case are they all equal. We also know that all the eigenvalues are real and positive valued for symmetric, positive definite autocorrelation matrices. The resulting spread of the eigenvalues causes the LMS algorithm to converge very slowly. Most practical applications require the filter to operate in the presence of highly correlated signals, which severely limits the performance of the LMS algorithm. With the quasi-Newton family of algorithms, we can compensate for the degradations caused by colored input signals on the adaptive process.

The eigenvalues are bounded by the minimum and maximum values of the input power spectrum. As the order of the 2-D adaptive filter increases, λ_{min} and λ_{max} quickly approach these limits. Once again, the 2-D autocorrelation matrix is $(N+1)^2$ by $(N+1)^2$ for an $(N+1)$ by $(N+1)$ filter, so the condition number, $\kappa = \lambda_{max}/\lambda_{min}$, can become very large very fast even for moderately sized 2-D filters. For some of the experiments that follow, we chose the 5 by 5, separable, FIR low-pass coloring filter whose 2-D frequency response is shown in Figure 2.1a).



a)



b)

Figure 2.1 a) Frequency response of the separable, 5x5, FIR, low-pass coloring filter, and b) a plot of the eigenvalues of the corresponding 9x9 and 49x49 autocorrelation matrices with condition numbers 445 and 37512.

It has a normalized cutoff frequency of 0.1 and was designed using a standard 1-D FIR filter design package. Shown in Figure 2.1b) are plots of the eigenvalues of (9 by 9) and (49 by 49) autocorrelation matrices constructed using this particular process. Each of these matrices corresponds to adaptive filters with regions of support of (3 by 3) and (7 by 7), respectively. The condition numbers for each of these cases are κ_9=445 and κ_{49}=37,512, clearly demonstrating the rapid ill-conditioning evident with 2-D filters.

All computer experiments were performed using 2-D system identification with an artificial noise floor of -100 dB added to the desired signal. Furthermore, double precision arithmetic is used unless specifically noted. The first set of experiments demonstrates the performance of the 2-D LMS filter with white and colored noise, and then compares the rate of convergence of the 2-D fast quasi-Newton algorithm with the same colored input. The fast quasi-Newton algorithm uses the biased autocorrelation lag estimate and the block Levinson algorithm, both from Section 2.3, with the input process being stationary. Figure 2.2 shows the second-order LMS filter converging to -100 dB in 300 iterations for the case with white input and to -60 dB in 7000 iterations for the colored noise case. In Figure 2.3 the fast quasi-Newton algorithm with colored noise converges to -100 dB in about 400 iterations. This is a fantastic improvement at the cost of using a O[N^3] algorithm instead of the O[N^2] LMS algorithm. Similar results can be seen in the two groups of plots in Figures 2.4, 2.5, 2.6 and Figures 2.7, 2.8, 2.9 for fourth-order and sixth-order FIR adaptive filters. In each case, the fast quasi-Newton algorithm succeeds in improving convergence performance in the presence of colored noise to nearly that of the LMS algorithm with white noise. This is clearly the anticipated result from earlier discussions. The LMS error gradient *approximation* and the autocorrelation *estimate* now limit the performance of the quasi-Newton algorithm. Again, similar results are shown in Figures 2.10, 2.11, 2.12, and 2.13.

Figure 2.14 shows convergence plots emphasizing the nonstationary behavior of the filter studied in Figures 2.2 and 2.3. Here the autocorrelation matrix estimate is initialized to the identity matrix, then the filter is suddenly submersed in the same low-pass colored noise field discussed above. For these two examples, we used horizontal scan lines 30 pixels long before retracing to

the next row. These plots show the initial rate of convergence with α=0.9 and α=0.99. The filter converges at a slightly slower rate now that the lag estimation algorithm is also in the process of converging. However, the lag coefficients are quickly identified, and the filter quickly approaches its optimum rate of convergence. Notice also that when the forgetting factor is reduced the algorithm's performance seems to improve. The filter converges faster with α=0.9 than with α=0.99. This indicates that the new filter has good nonstationary convergence properties, but more extensive work may be needed under many different circumstances.

In each experiment using the fast quasi-Newton algorithm, the run length block size is at least $(N+1)^2$ iterations. Therefore, the autocorrelation matrix at the end of each block is used to process the next consecutive block of data. However, the lag estimates are continually updated at each index point. Also, the exponential weighting factor was nominally chosen to be 0.97 for each case except as explicitly stated above for the nonstationary experiments.

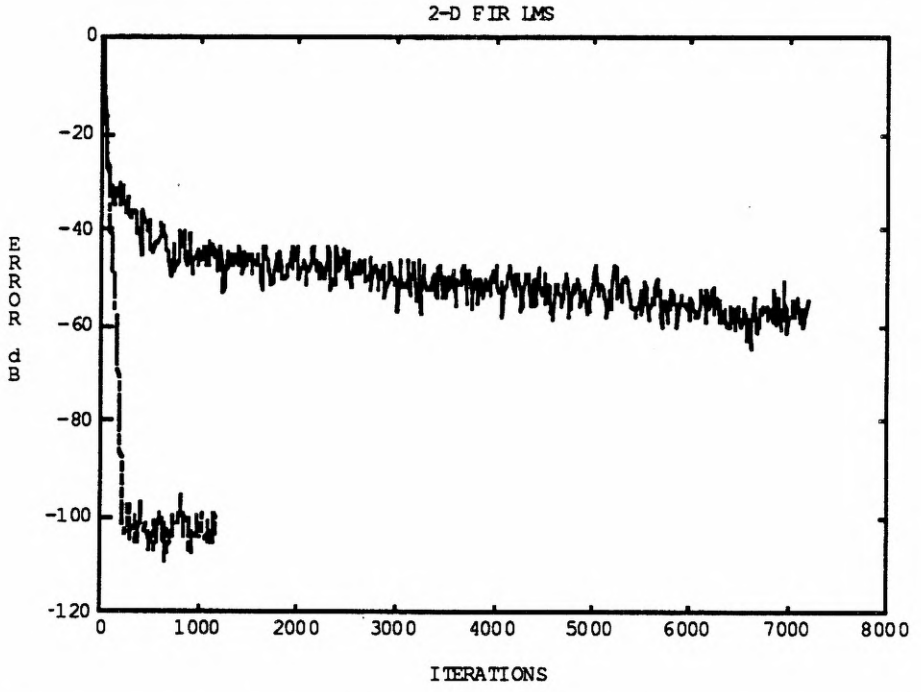


Figure 2.2 Convergence plot for a 2-D, 3x3, FIR, LMS adaptive filter in system identification with white input (below) and colored input (above).

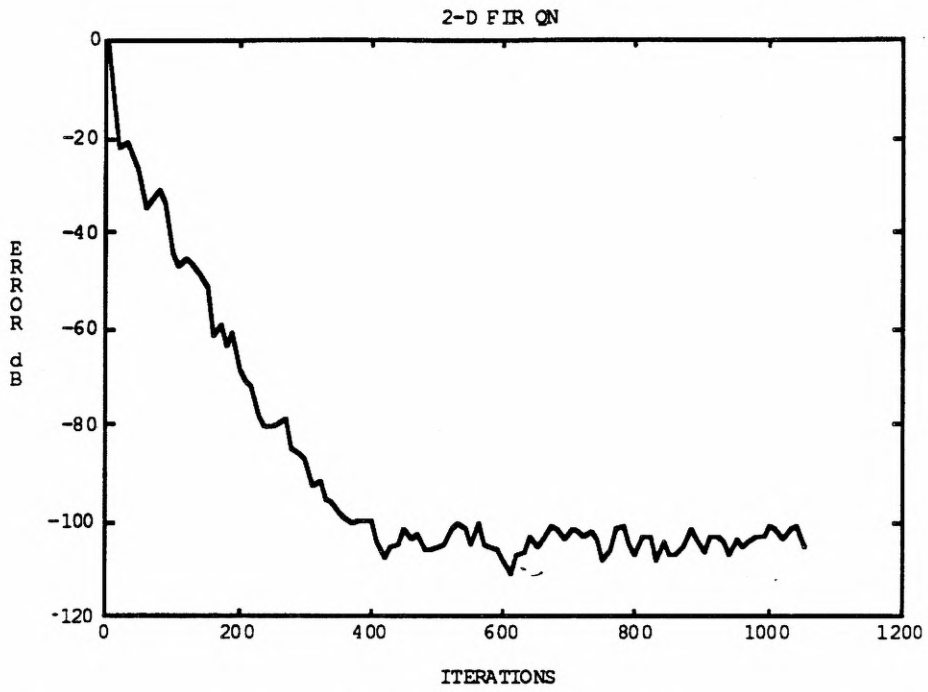


Figure 2.3 Convergence plot for a 2-D, 3x3, FIR, fast Quasi-Newton adaptive filter in system identification with colored noise.

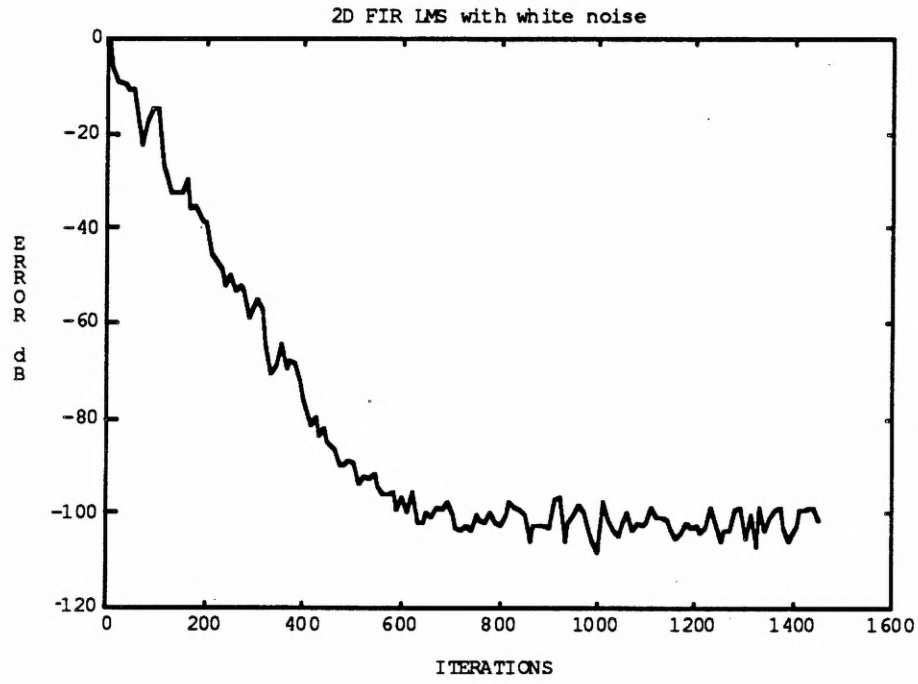


Figure 2.4 Convergence plot for a 2-D, 5x5, FIR, LMS adaptive filter in system identification with white input noise.

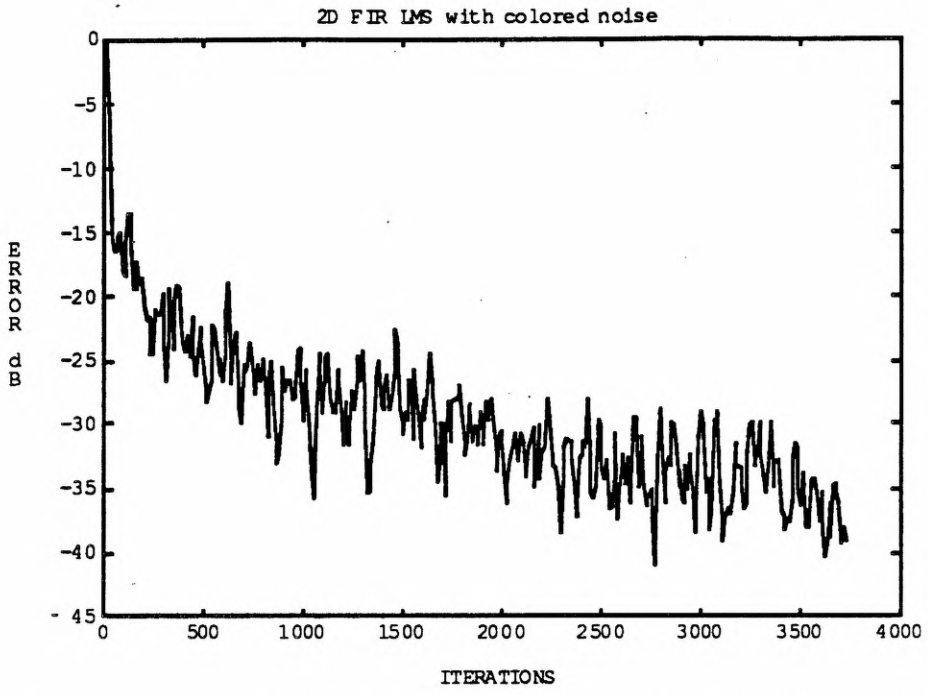


Figure 2.5 Convergence plot for a 2-D, 5x5, FIR, LMS adaptive filter in system identification with colored input noise.

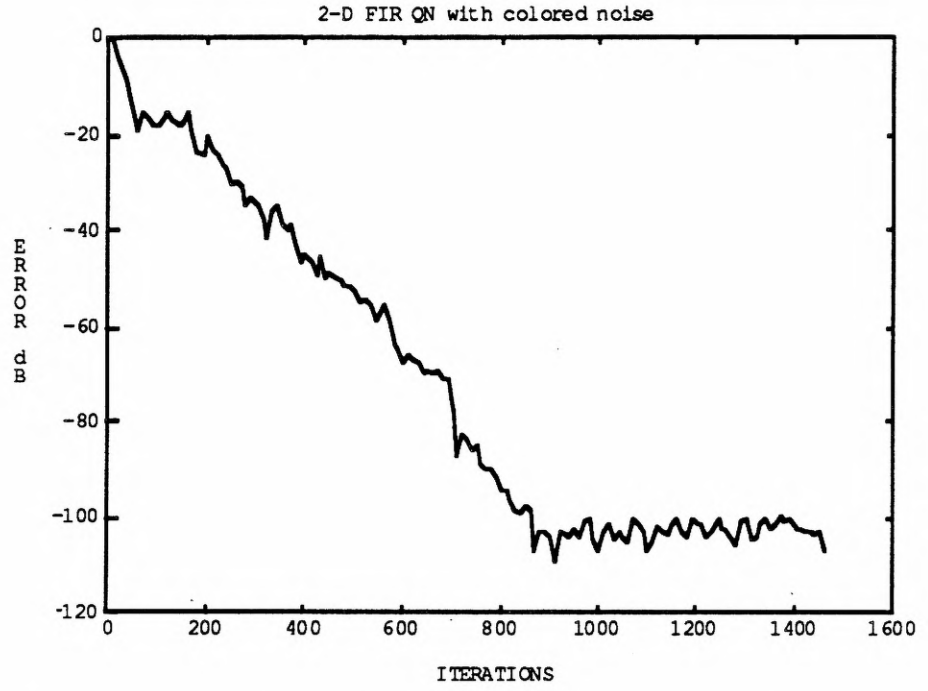


Figure 2.6 Convergence plot for a 2-D, 5x5, FIR, FQN adaptive filter in system identification with colored input noise.

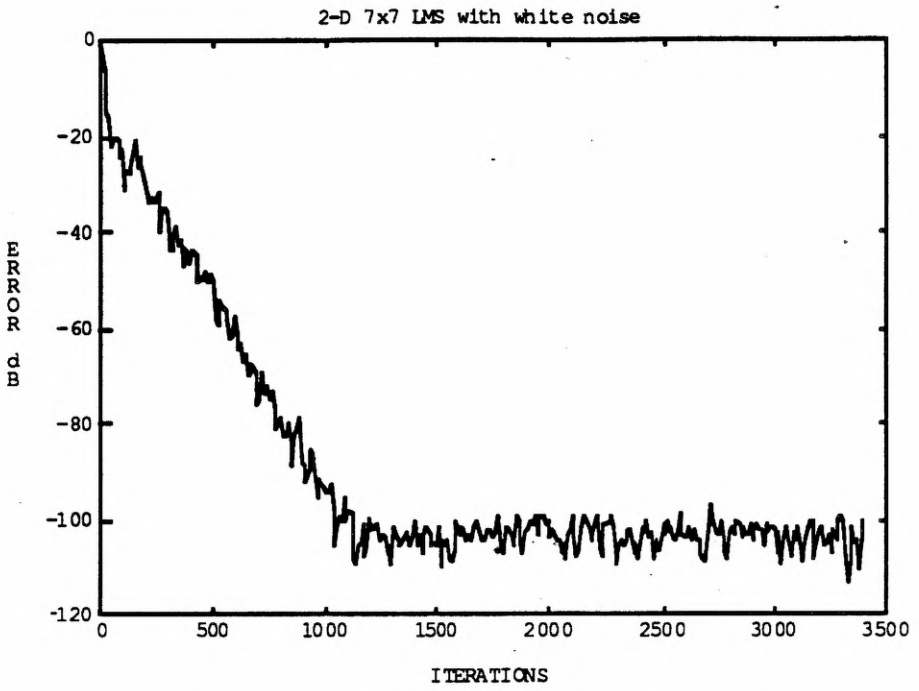


Figure 2.7 Convergence plot for a 2-D, 7x7, FIR, LMS adaptive filter in system identification with white input noise.

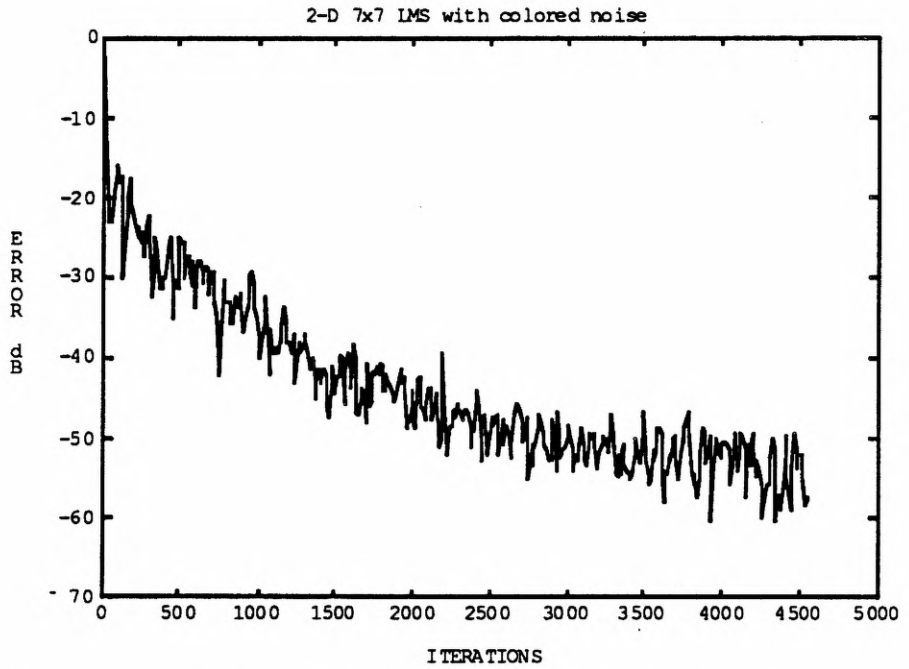


Figure 2.8 Convergence plot for a 2-D, 7x7, FIR, LMS adaptive filter in system identification with colored input noise.

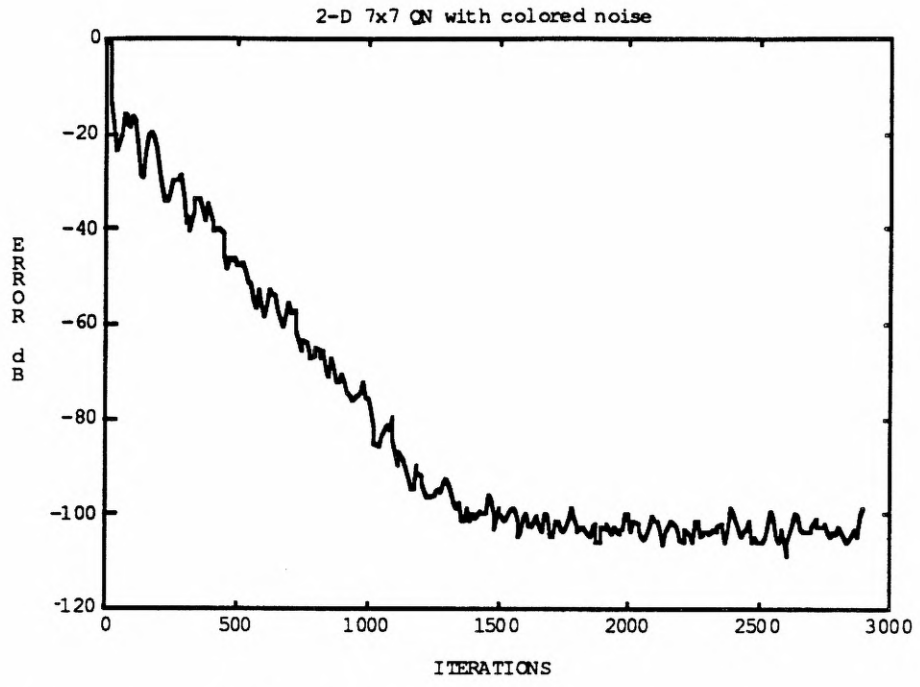


Figure 2.9 Convergence plot for a 2-D, 7x7, FIR, FQN adaptive filter in system identification with colored input noise.

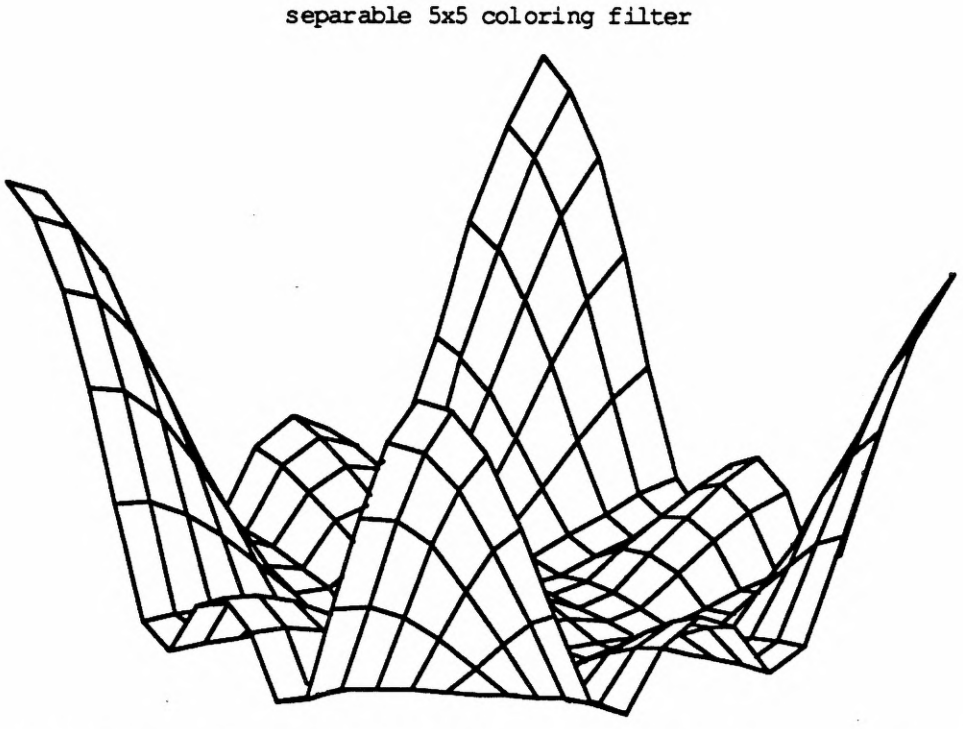


Figure 2.10 Frequency response of the separable, 5x5, FIR, high pass coloring filter.

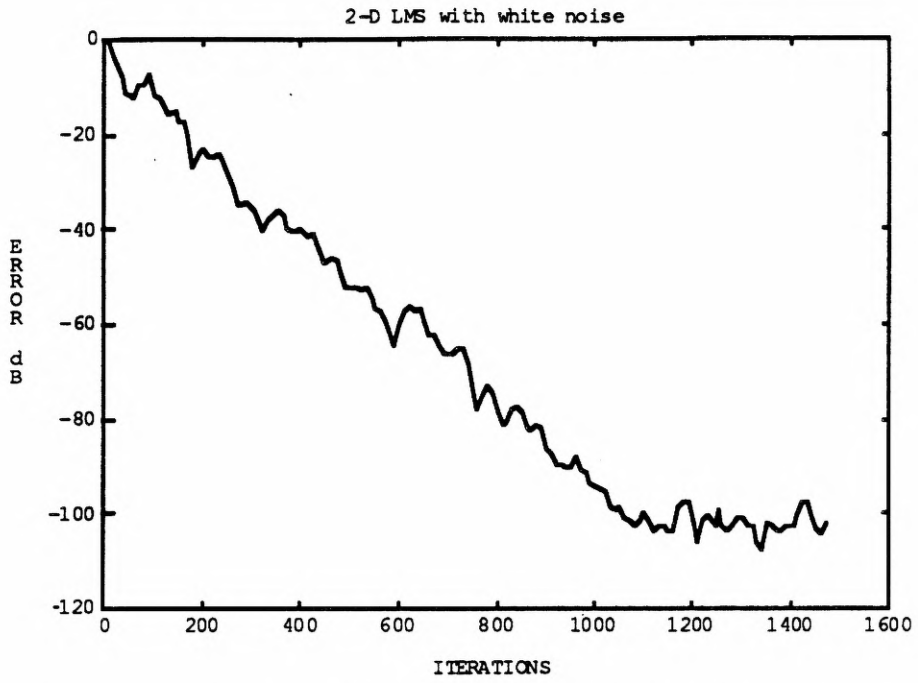


Figure 2.11 Convergence plot for a 2-D, 7x7, FIR, LMS adaptive filter in system identification with white input noise.

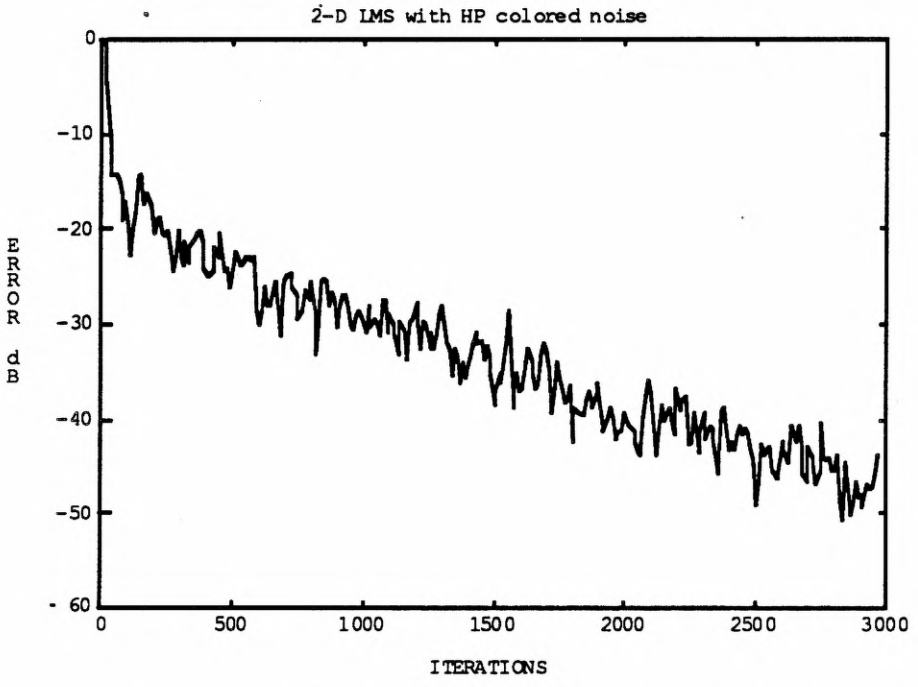


Figure 2.12 Convergence plot for a 2-D, 7x7, FIR, LMS adaptive filter in system identification with high pass colored input noise.

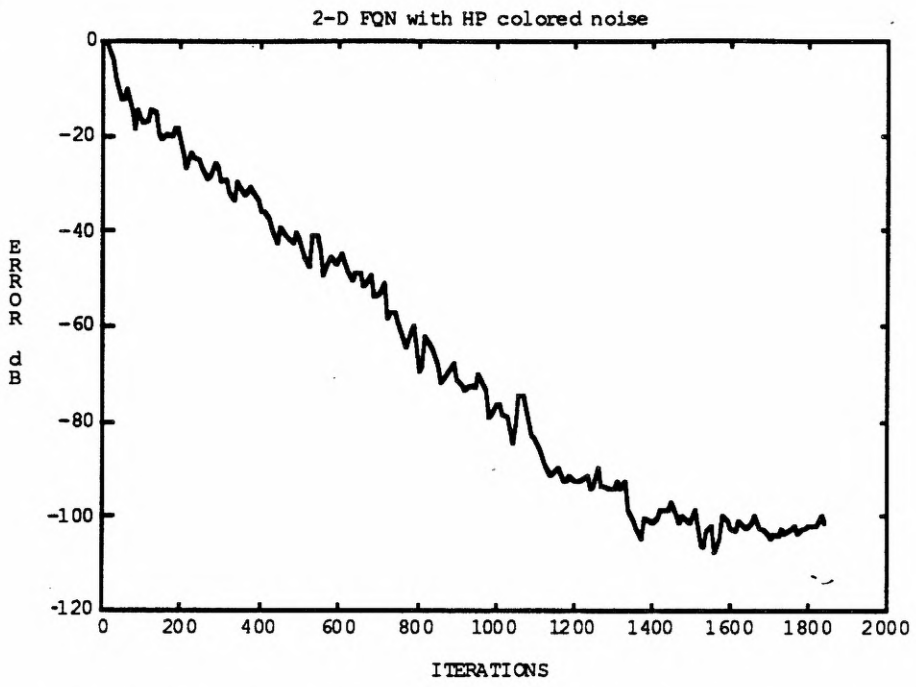


Figure 2.13 Convergence plot for a 2-D, 7x7, FIR, FQN adaptive filter in system identification with high-pass colored input noise.

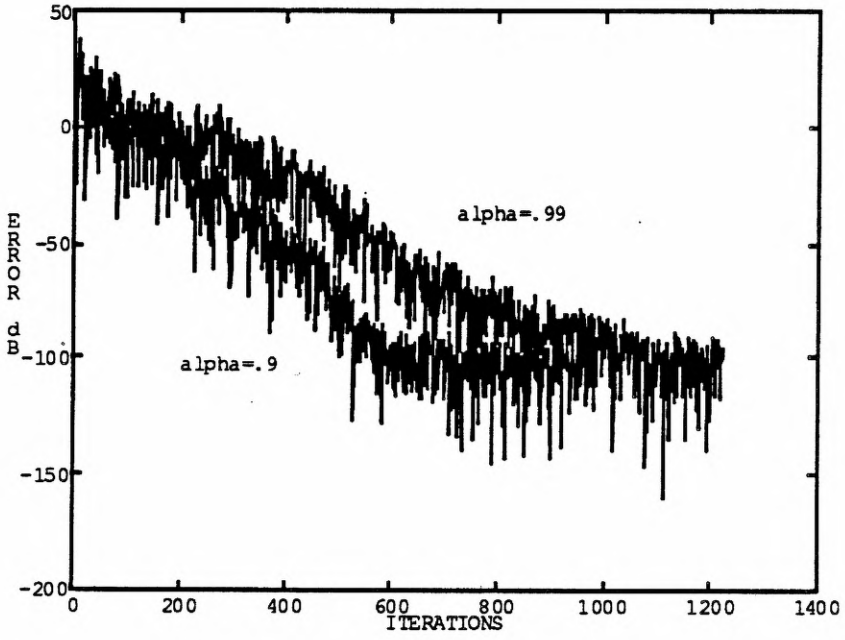


Figure 2.14 Nonstationary convergence plots for the same 2-D, 3x3, FIR, FQN adaptive filter from Figure 2.3 with the same low-pass colored input signal.

2.5 Comparison to the 2-D Transform Domain Adaptive Filter

Other successful 1-D FIR algorithms have been extended to 2-D filters. Transform domain adaptive algorithms are also well-suited to 2-D signal processing. Orthogonal transforms with power normalization can be used to accelerate the convergence of an adaptive filter in the presence of a colored input signal. Transform domain adaptive filters first appeared in papers by Dentino et al. [73], Narayan et al. [74], and others. Marshall et al. [75] examined several different transforms for 1-D filters, and the work was extended to 2-D in [76] by Howard.

The TDAF structure is shown in Figure 2.15 with the corresponding (possibly complex) LMS algorithm [1],[2],[77] given as

$$\boldsymbol{h}_{k+1}(m_1,m_2) = \boldsymbol{h}_k(m_1,m_2) + 2\,\mu\, e(n_1,n_2)\, \Lambda_x^{-2}\boldsymbol{x}_k^*(n_1,n_2) \tag{2.47}$$

where $\boldsymbol{x}_k(n_1,n_2)$ is the column-ordered vector formed by premultiplying the input-column ordered vector $\boldsymbol{u}_k(n_1,n_2)$ by the 2-D unitary transform $\boldsymbol{T}$.

$$\boldsymbol{x}_k(n_1,n_2) = \boldsymbol{T}\,\boldsymbol{u}_k(n_1,n_2) \tag{2.48}$$

Channel normalization results from including $\Lambda_x^2 = \mathrm{diag}[\sigma_x^2(0,0)\ \sigma_x^2(1,0)\ \ldots\ \sigma_x^2(N,N)]$ in (2.47) where $\sigma_x^2(n_1,n_2) \approx \mathrm{E}[|\,\boldsymbol{x}(n_1,n_2)\,|^2]$. Ideally, the Karhunen-Loeve Transform (KLT) is used to achieve optimal convergence, but this requires a priori knowledge of the input statistical properties. The KLT corresponding to the input autocorrelation matrix $\boldsymbol{R}_u$ is constructed using as rows of $\boldsymbol{T}$ the orthonormal eigenvectors of $\boldsymbol{R}_u$. Therefore, with unitary $\boldsymbol{Q}_u^{\mathrm{H}}=[\boldsymbol{q}_1\ \ldots\ \boldsymbol{q}_M 2]$ and $\Lambda_u=\mathrm{diag}[\lambda_1\ \ldots\ \lambda_M 2]$ ($M=N+1$ for convienience), the unitary similarity transformation is $\boldsymbol{R}_u=\boldsymbol{Q}_u^{-1}\Lambda_u\boldsymbol{Q}_u$, and the KLT is given by $\boldsymbol{T}=\boldsymbol{Q}_u$. However, since the statistical properties of the input process are usually unknown and time varying, the KLT cannot be implemented in practice.

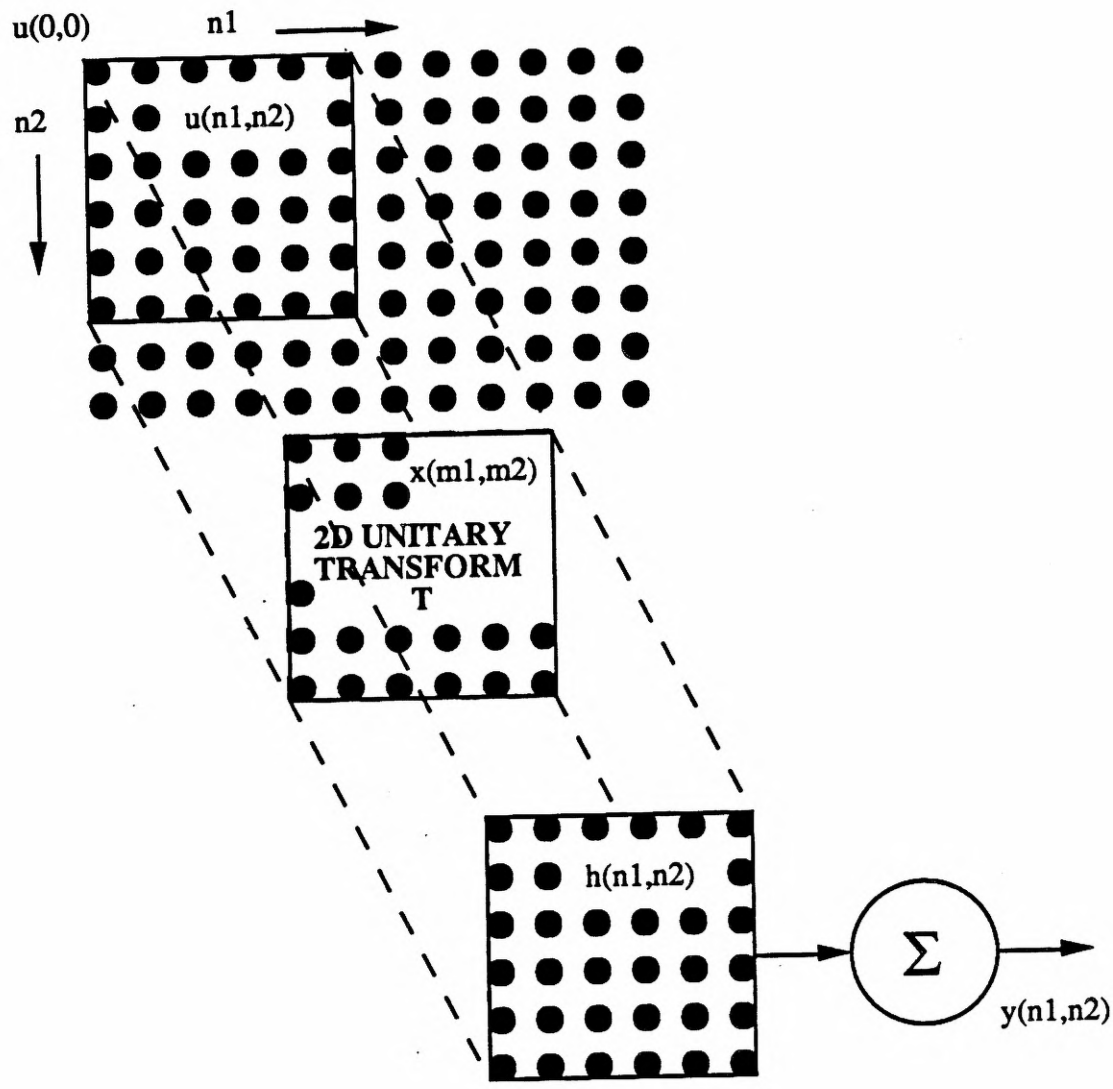


Figure 2.15 Two-dimensional transform domain adaptive filter structure.

Researchers have found that many fixed transforms do provide good orthogonalization for a wide class of input signals. Those include the Discrete Fourier Transform (DFT or FFT), the Discrete Cosine Transform (DCT), and the Walsh Hadamard Transform (WHT). For example, the DFT provides only approximate channel decorrelation since it is well-known that a "sliding" DFT implements a parallel bank of overlapping band-pass filters with center frequencies evenly distributed over the interval $[0,2\pi]$. Furthermore, the DFT (or FFT) is hampered by the fact that it

requires complex arithmetic. It is still a very effective method of orthogonalization which we compare here to the 2DFQN algorithm.

The convergence plots in Figure 2.16 show the comparison between the 2DFQN, the 2DTDAF and the simple 2DLMS with the same fourth-order low-pass coloring filter from Section 2.4. The adaptive filter is second order, and the 2DFQN algorithm, as expected, outperforms the 2DTDAF. The 2DFQN algorithm is effectively attempting to estimate the KLT on-line so that, while not able to perfectly orthogonalize the training signal, it does offer improved convergence over that of the fixed transform algorithm. Similar results appear in Figure 2.17 with the same coloring filter and a fourth-order adaptive filter.

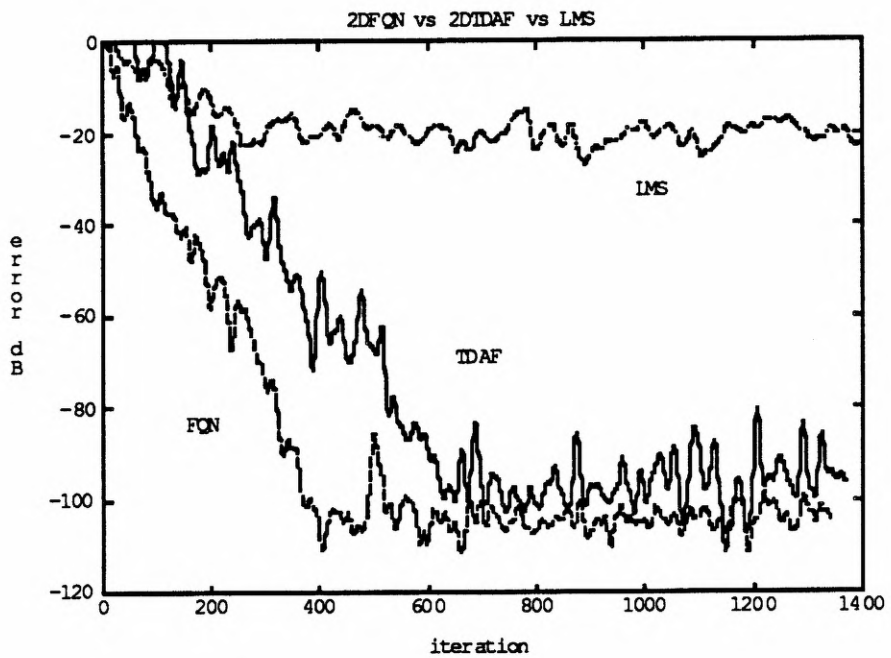


Figure 2.16 Convergence plot for a 2-D, 3x3, FIR, FQN adaptive filter in system identification with low-pass colored input. Corresponding convergence plots are shown for the 2DTDAF-DFT and the 2DLMS.

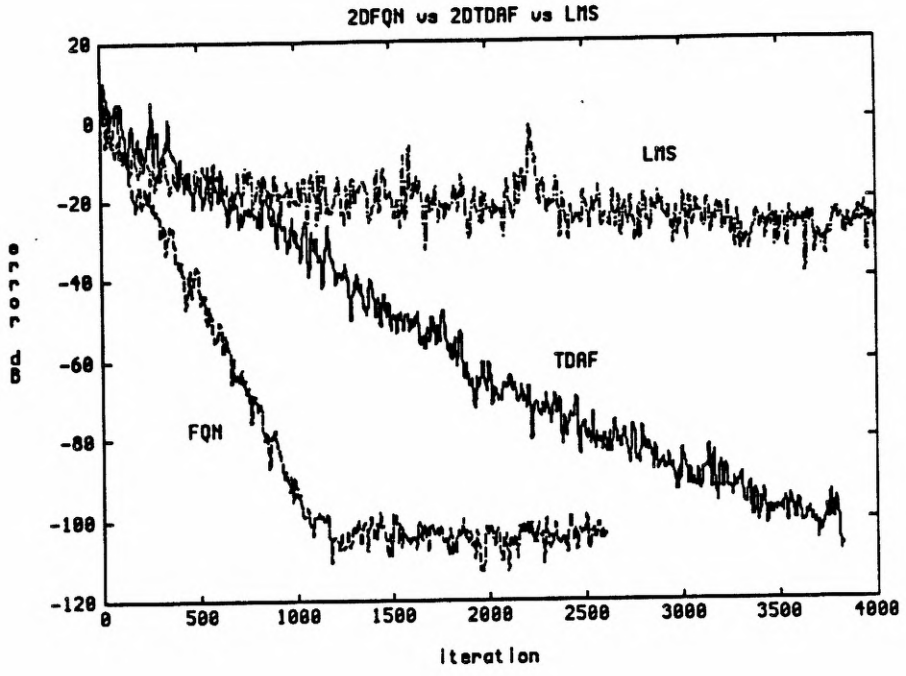


Figure 2.17 Convergence plot for a 2-D, 5x5, FIR, FQN adaptive filter in system identification with low-pass colored input. Corresponding convergence plots are shown for the 2DTDAF-DFT and the 2DLMS.

2.6 The 2-D Recursive Least Squares Algorithm

For completeness we present the least squares (LS) and recursive least squares (RLS) solution methods for 2-D FIR adaptive filtering. These results also serve as a benchmark for comparison to other structures and algorithms which we consider. The least squares algorithm is a prerequisite for the development of the RLS algorithm, and both are unique in that they exactly solve the system of equations which result from minimizing the sum of squared errors over a region of data [78]. The extension to 2-D FIR filtering is straightforward with the standard system parameters as previously defined:

$$\boldsymbol{u}(n_1,n_2) = \text{input} = \begin{bmatrix} u(n_1,n_2) \\ \vdots \\ u(n_1\text{-}N,n_2\text{-}N) \end{bmatrix} \tag{2.49}$$

$$\boldsymbol{h}(n_1,n_2) = \text{2-D filter} = \begin{bmatrix} h(0,0) \\ \vdots \\ h(N,N) \end{bmatrix} \tag{2.50}$$

$y(n_1,n_2)$ = 2-D filter output

$d(n_1,n_2)$ = desired output

$$J_{ss} = \text{cost function} = \sum_{\mathfrak{R}} | d(n_1,n_2) - y(n_1,n_2) |^2 \tag{2.51}$$

The region $\mathfrak{R}$ over which the minimization occurs can be defined as any arbitrarily shaped block of 2-D data. Since all signals are real and $y(n_1,n_2)=\boldsymbol{u}^{\mathrm{T}}(n_1,n_2)\boldsymbol{h}(n_1,n_2)$, the cost function can be expressed as

$$J_{ss} = \sum_{\mathfrak{R}} [d(n_1,n_2)^2\text{-}2d(n_1,n_2)\boldsymbol{u}^{\mathrm{T}}(n_1,n_2)\boldsymbol{h}(n_1,n_2)+\boldsymbol{h}^{\mathrm{T}}(n_1,n_2)\boldsymbol{u}(n_1,n_2)\boldsymbol{u}^{\mathrm{T}}(n_1,n_2)\boldsymbol{h}(n_1,n_2)] \tag{2.52}$$

Optimizing with respect to the filter coefficients, we have

$$\frac{\partial J_{ss}}{\partial \boldsymbol{h}(n_1,n_2)} = \sum_{\mathfrak{R}} [-2d(n_1,n_2)\boldsymbol{u}(n_1,n_2)+2\boldsymbol{u}(n_1,n_2)\boldsymbol{u}^{\mathrm{T}}(n_1,n_2)\boldsymbol{h}(n_1,n_2)] \tag{2.53}$$

Now define

$$\boldsymbol{p} = \sum_{\mathfrak{R}} d(n_1,n_2)\boldsymbol{u}(n_1,n_2) = \text{deterministic cross correlation vector} \tag{2.54}$$

$$\boldsymbol{R} = \sum_{\mathfrak{R}} \boldsymbol{u}(n_1,n_2)\boldsymbol{u}^{\mathrm{T}}(n_1,n_2) \;=\; \text{deterministic auto correlation matrix} \tag{2.55}$$

Then $0=-\boldsymbol{p}+\boldsymbol{R}\boldsymbol{h}$ or $\boldsymbol{p}=\boldsymbol{R}\boldsymbol{h}$, giving the least squares filter as $\boldsymbol{h}=\boldsymbol{R}^{-1}\boldsymbol{p}$.

The matrix inversion lemma, $(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{C}\boldsymbol{D})^{-1}=\boldsymbol{A}^{-1}-\boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B}+\boldsymbol{C}^{-1})^{-1}\boldsymbol{D}\boldsymbol{A}^{-1}$, can be used to solve this equation recursively resulting in a 2-D RLS algorithm. Now let $\boldsymbol{A}=\boldsymbol{R}_k$, $\boldsymbol{B}=y$, $\boldsymbol{C}=1$, $\boldsymbol{D}=\boldsymbol{u}^{\mathrm{T}}$; then using the least squares solution, $\boldsymbol{h}_{k+1}=\boldsymbol{R}_{k+1}^{-1}\boldsymbol{p}_{k+1}$, with $\boldsymbol{p}_{k+1}=\boldsymbol{p}_k+d(n_1,n_2)\boldsymbol{u}(n_1,n_2)$ and $\boldsymbol{R}_{k+1}=\boldsymbol{R}_k+\boldsymbol{u}(n_1,n_2)\boldsymbol{u}^{\mathrm{T}}(n_1,n_2)$, we obtain the 2DRLS algorithm just as in the 1D case

$$\boldsymbol{h}_k = \boldsymbol{R}_k^{-1}\boldsymbol{p}_k \tag{2.56a}$$

$$\boldsymbol{z}_k = \boldsymbol{R}_k^{-1}\boldsymbol{u}_k \tag{2.56b}$$

$$q = \boldsymbol{u}_k^{\mathrm{T}}\boldsymbol{z}_k \tag{2.56c}$$

$$\boldsymbol{h}_{k+1} = \boldsymbol{h}_k + \frac{[d(n_1,n_2)-y_k]\boldsymbol{z}_k}{1+q} \tag{2.56d}$$

The region $\mathfrak{R}$, the indexing scheme, and the windowing method define the mapping between the image index and the vector index above. Since the familiar LS and RLS solutions are directly applicable to this problem all of the well-known properties also hold true for 2-D filters. The number of parameters is $(N+1)^2$ instead of $N+1$, so we expect the 2-D RLS algorithm to require $(N+1)^2$ iterations to converge. In addition, the computational complexity is $O[N^4]$, as compared to $O[N^2]$ for the 2-D LMS algorithm. We also expect the 2DRLS algorithm to have numerical difficulties and poor tracking performance. From the equations above we see that the 2DRLS is a member of the general quasi-Newton class of algorithms. Fast RLS algorithms do exist and have been extended to 2-D adaptive filtering. For example, a 2-D geometrical fast transversal filter was developed in [32] as a direct extension of the work done by Alexander [79] for 1-D filters. In Figure 2.18 we present an example of a 2-D RLS adaptive filter in the familiar system identification configuration. This example shows a 3x3 FIR adaptive filter identifying a matched-order fixed

filter with a pseudo-white input signal. The RLS algorithm converges in about nine iterations, which is what we expect since the filter has nine coefficients.

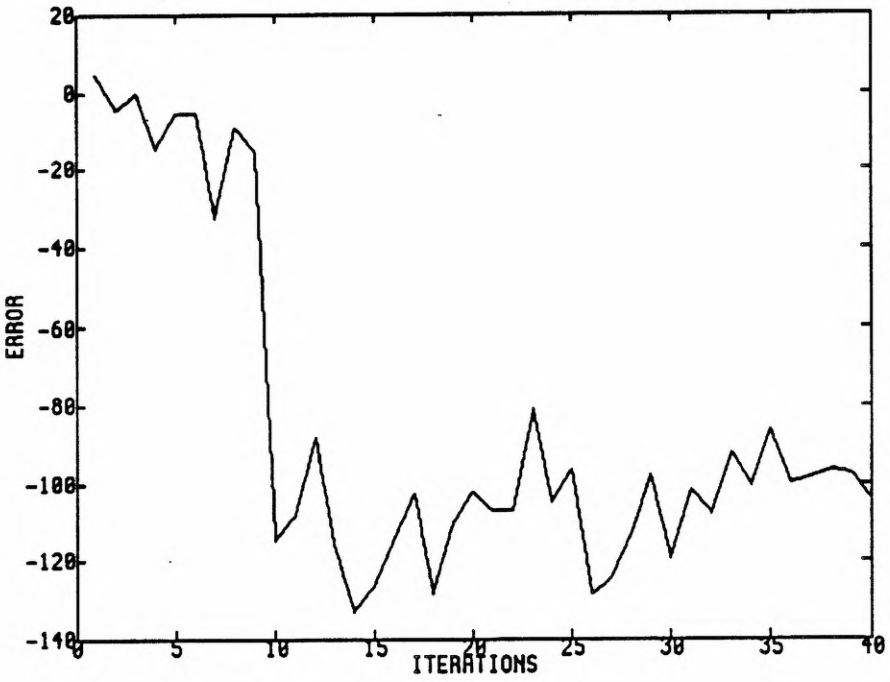


Figure 2.18 Convergence plot for a second-order 2-D FIR RLS adaptive filter.

2.7 The McClellan Transformation Structure

A 2-D adaptive filter structure based on the McClellan transformation (MCT) design technique for 2-D FIR filters has been proposed as a means of improving the performance and computational requirements of 2-D adaptive signal processing [14]-[17]. It was shown that the MCT structure can be constrained with a priori information about the contour shapes of the frequency response so that the resulting adaptive algorithm is computationally efficient and rapidly converging. The MCT design technique uses a 1-D prototype filter along with a transformation

function to define the characteristics of the 2-D FIR frequency response. The frequency response of a 1-D zero phase FIR filter is

$$H(\omega) = \sum_{n=-N}^{N} h(n)\exp(-j\omega n) \tag{2.57}$$

$$= h(0) + \sum_{n=1}^{N} h(n)[\exp(-j\omega n) + \exp(j\omega n)] \tag{2.58}$$

$$= \sum_{n=0}^{N} a(n)\cos(\omega n) \tag{2.59}$$

where

$$a(n) = \begin{cases} h(n) & n = 0 \\ 2h(n) & 1 \le n \le N \end{cases} \tag{2.60}$$

The function $\cos(\omega n)$ may be expressed as the n^{th} Chebyshev polynomial in the variable $\cos(\omega)$ so that (2.59) becomes

$$H(\omega) = \sum_{n=0}^{N} a(n)T_n[\cos(\omega)] \tag{2.61}$$

The 1-D prototype filter is mapped to a 2-D filter with a 2-D transformation function $F(\omega_1,\omega_2)$.

$$H(\omega) = \sum_{n=0}^{N} a(n)T_n[F(\omega_1,\omega_2)] \tag{2.62}$$

The transformation function itself must have a 2-D real-valued, zero phase response bounded between -1 and 1. The overall transfer function will have the same contour shapes as the transformation function, F. For the case of a $(2P+1)$x$(2P+1)$ transformation function and a size $2N+1$ prototype filter, the MCT filter will have a $(2NP+1)$ by $(2NP+1)$ region of support. The most common choice for the transformation function is

$$F(\omega_1,\omega_2) = A + B\cos(\omega_1) + C\cos(\omega_2) + D\cos(\omega_1)\cos(\omega_2) \tag{2.63}$$

The filter design problem is now modular with the contour parameters (A,B,C,D) controlling the contour shapes, and the prototype coefficients, $a(n)$, controlling the magnitude of the 2-D frequency response at each particular contour. Figure 2.19 shows contour plots for circular and fan-shaped contours.

The transformation parameters for a filter with a frequency response with circular-shaped contours are (A, B, C, D)=(-0.5, 0.5, 0.5, 0.5), and for fan-shaped frequency contours we have (A, B, C, D)=(0, 0.5, -0.5, 0). We can design an elliptical MCT filter with transformation parameters (A, B, C, D)=(0, 0.05, 0.722, 0.228).

An efficient implementation of the MCT structure based on the Chebyshev recurrence formula exists, so that the computational complexity is ($5N$+1) multiplications per output sample. That relation is

$$T_n[x] = 2\,x\,T_{n-1}[x] - T_{n-2}[x] \tag{2.64}$$

so that the structure in Figure 2.20 results [14].

Now if the contour shape requirements are known a priori, then the transformation parameter set, (A, B, C, D), can be chosen and fixed so that the adaptive parameter set consists exclusively of the tap weights, $a(n)$. The resulting constrained structure represents a reduced parameterization model of the desired 2-D response, and the simple gradient-based adaptive algorithm is efficient, rapidly converging, and numerically robust. Applying the LMS algorithm to the MCT adaptive filter gives

$$a_{k+1}(n) = a_k(n) + 2\,\mu\,e(n_1,n_2)\,s_n(n_1,n_2) \tag{2.65}$$

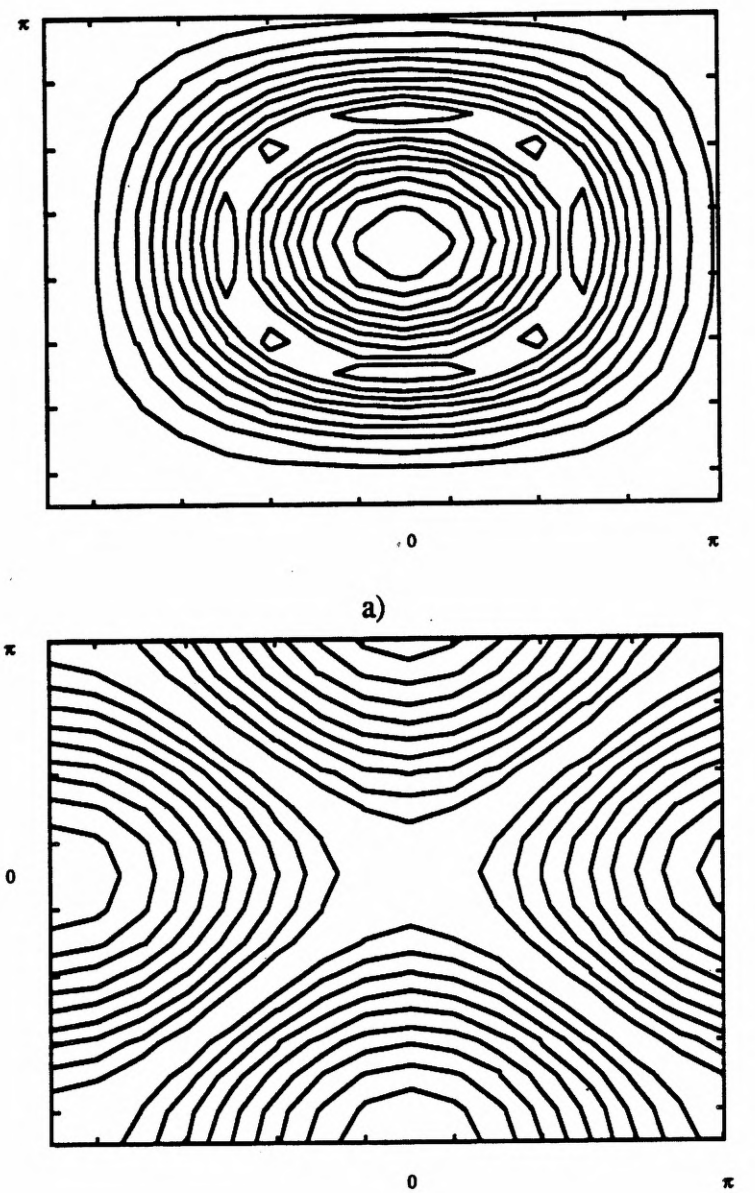


Figure 2.19 a) Circular and b) fan-shaped frequency domain contours.

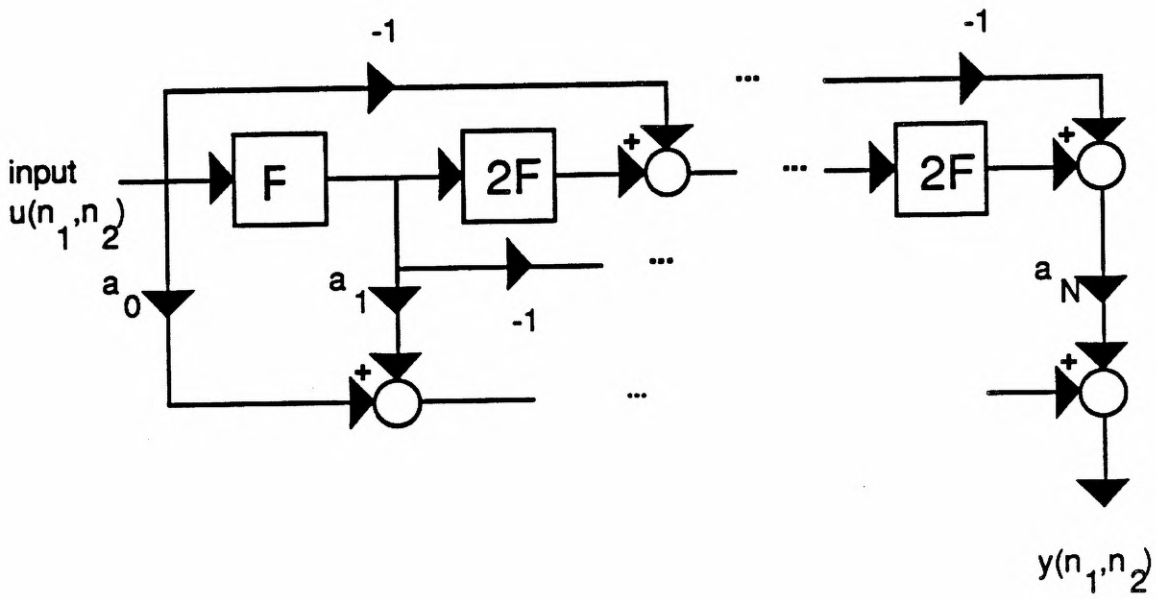


Figure 2.20 The McClellan transformation structure.

where $s_n(n_1,n_2)$ is the signal after the n^{th} stage. Since there are now only N+1 parameters, complexity of the adaptive algorithm is O[N]. Convergence results show that the MCT filter adapts much faster that the 2-D direct form LMS filter. The observed rate of convergence lies between those of a size N+1 1-D LMS filter and a (2N+1)x(2N+1) 2-D LMS filter. Mathematically, the MCT LMS filter is equivalent to a length N+1 1-D adaptive filter with a colored input signal. At this point, the adaptive process may be accelerated using an orthogonal transform or a recursive least squares algorithm. Using the 1-D FFT just before the tap weights, $a(n)$, along with power normalization increases the rate of convergence substantially.

Finally, the tangential filter can be adapted by including the contour parameters (A,B,C,D) in the adaptive parameter set. This enables the structure to model a wider class of filter responses with different contour shapes. The additional flexibility comes at the expense of a more complex

coefficient update since the MCT filter error is a nonlinear function of each transformation parameter. Experimental results indicate best convergence can be obtained using a two-stage approach [15]. First, the MCT adaptive filter is allowed to identify the contours with a relatively large convergence factor for the transformation parameter set and a small convergence factor for the tap weights. Then the process is reversed so that the transformation parameters adapt slowly once they are nearly identified, and the tap weights are allowed to adapt quickly with a larger convergence factor.

A convergence analysis for the McClellan transformation structure is straightforward. From Equation (2.62), we obtain

$$y(n_1,n_2) = \sum_{n=0}^{N} a(n)\mathrm{FT}^{-1}\{X(\omega_1,\omega_2)T_n[F(\omega_1,\omega_2)]\} \tag{2.66}$$

where FT^{-1} is the inverse discrete time Fourier transform. By defining the intermediate stage signal $s_n(n_1,n_2)=x(n_1,n_2)**t_n(n_1,n_2)$, Equation (2.66) becomes

$$y(n_1,n_2) = \sum_{n=0}^{N} a(n)s_n(n_1,n_2) \tag{2.67}$$

Now we define the tap weight vector as $\boldsymbol{a}=[a_0, \ldots, a_N]^T$ and the intermediate signal vector as $\boldsymbol{s}=[s_0(n_1,n_2), \ldots, s_N(n_1,n_2)]^T$. The mean-squared error analysis now gives

$$\mathrm{MSE} = \mathrm{E}\{d^2(n_1,n_2)\} - 2\boldsymbol{p}_s^T\boldsymbol{a} + \boldsymbol{a}^T\boldsymbol{R}_s\boldsymbol{a} \tag{2.68}$$

with

$\boldsymbol{p}_s = \mathrm{E}\{d(n_1,n_2)\boldsymbol{s}\}$ = intermediate cross-correlation vector

$\boldsymbol{R}_s = \mathrm{E}\{\boldsymbol{s}\boldsymbol{s}^T\}$ = intermediate autocorrelation matrix.

The Wiener solution is easily found to be $\boldsymbol{a}^* = \boldsymbol{R}_s^{-1}\, \boldsymbol{p}_s$, with the minimum mean-squared error equal to $E\{d^2(n_1,n_2)\}-\boldsymbol{p}_s^T\boldsymbol{a}^*$. It also can be shown easily that the LMS algorithm is unbiased with the expected value of the tap weight vector being the Wiener solution. Analysis of the LMS algorithm for this case $\{a_{k+1}(n) = a_k(n) + 2\,\mu\, e(n_1,n_2)\, s_n(n_1,n_2)\}$ is similar to that of the standard 1-D LMS case. After translating and rotating the tap weight coordinate vector and taking expected values, the LMS algorithm for the McClellan transformation filter gives a familiar difference equation

$$E\{\boldsymbol{c}_{j+1}\} = [\mathbf{I}-2\mu\Lambda]\, E\{\boldsymbol{c}_j\} \tag{2.69}$$

where $\boldsymbol{c}_j$ is the new tap weight vector and Λ is the diagonalized intermediate autocorrelation matrix. The solution of (2.69) is

$$E\{\boldsymbol{c}_j\} = [\mathbf{I}-2\mu\Lambda]^j\, E\{\boldsymbol{c}_0\}, \tag{2.70}$$

which converges to zero if $1/\lambda_{max} > \mu > 0$. Now we have an adaptive system with $(N+1)$ modes, and the eigenvalues of $\boldsymbol{R}_s$ govern convergence exactly as in the 1-D case. From Widrow and Stearns [1], we know that the rate of convergence depends on the number of adaptive modes as well as the relative values of the eigenvalues. We can therefore conclude that the MCT structure converges at a rate similar to that of a 1-D FIR adaptive filter of length N with a colored input process. An example appears in Figure 2.21 with a convergence plot for a sixth-order MCT filter with circular contours. The learning curve for a direct form, FIR, LMS filter accompanies this example for comparison. The input signal for both filters is white.

Least squares and recursive least squares algorithms can be formulated after writing the MCT output as $y\,(n_1,n_2)=\boldsymbol{a}^T\boldsymbol{s}(n_1,n_2)$. We then can use this expression in a standard least squares minimization which gives an optimal tap weight vector, $\boldsymbol{a}^*$, in terms of the deterministic

autocorrelation matrix, $\boldsymbol{R}_s$, and the deterministic cross-correlation vector, $\boldsymbol{p}_s$, just as in Section 2.6. We simply use the intermediate signal $s(n_1,n_2)$ in place of the column-ordered input signal $u(n_1,n_2)$, and the algorithm can be applied as given in Equation (2.56). Now we have a 2-D adaptive filter which we expect to converge in about N iterations with $O(N^2)$ complexity for a linear phase filter with a $(2N+1)$x$(2N+1)$ region of support. This algorithm will display the same numerical sensitivity problems, which are known to exist for all RLS algorithms.

In summary, the MCT adaptive filter provides rapid convergence, low filter complexity, low adaptive algorithm complexity, and good sensitivity. Drawbacks which must be considered include reduced structural flexibility and increased storage requirements. For example, the Chebyshev implementation requires about N-i+1 rows of storage for each i^{th} stage. Also, some a priori statistical knowledge may be required before the filter can be used. The structure does offer promise for real-time video processing. The high video sampling rate (14 MHz) demands computational efficiency, and the MCT filter offers 2-D processing capability with 1-D computational costs. High speed, parallel processing DSP products are starting to appear on the market which enable some of the techniques discussed in this chapter to be realizable in typical real-time applications.

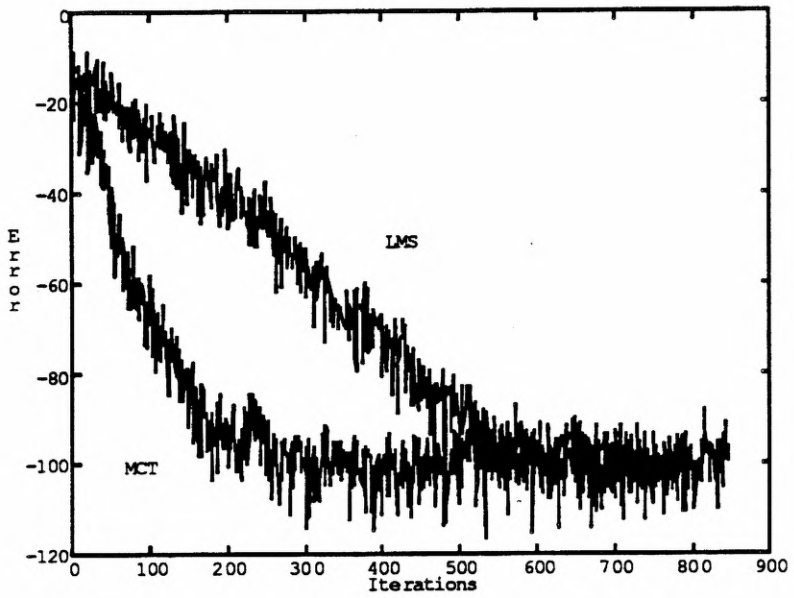


Figure 2.21 Learning curves for sixth-order MCT and direct form LMS filters with white input

CHAPTER 3
TWO-DIMENSIONAL IIR STRUCTURES AND ALGORITHMS

3.1 The Output Error Formulation

Infinite impulse response filters are useful in 1-D as well as 2-D signal processing applications because they offer reduced computational complexity with the reduced parameter set, yet increased modeling flexibility because of the recursive nature of the structure. For these reasons there has been much interest in 1-D IIR adaptive filtering in the research community, although IIR adaptive filters have yet to find widespread acceptance in industrial applications. Well-known difficulties such as slow convergence and the possible existence of multimodal error surfaces plague recursive adaptive filters, but many adaptive filtering applications are well-suited for IIR structures so that interest in this topic remains high. Later in this chapter (Section3.6) we present an investigation into the properties of the MSE surface of 2-D IIR adaptive filters. Some results are available for low-order filters in spite of the mathematical difficulties of the problem. Furthermore, 2-D data rates and computational requirements are extremely large so that the allure of IIR filters warrants investigation into the feasibility of using 2-D adaptive IIR filters. Typical video data rates restrict any digital processing so that only very small filters and rather crude algorithms may be used. In other instances, an inverse filter response may be required, which can only be approximated with an FIR filter.

Two-dimensional recursive adaptive filtering is similar in principle to iterative 2-D IIR filter design techniques [23]. Although the mean-squared error expression is nonlinear in the filter parameters, we can approach the problem using familiar adaptive processing techniques. As in FIR adaptive filtering, the simplest approach is to begin with a gradient-search algorithm seeking to minimize an error cost function. More elaborate algorithms can compensate for the effects of input

signal coloring on the rate of filter convergence. The steepest descent method can then be accentuated by including the Hessian matrix in the update relation. The Newton-Raphson method, for example, exploits first- and second-order statistics in order to traverse a more direct path to the optimal solution giving improved performance for cases in which the error surface can be approximated to be quadratic. We will concentrate on autoregressive moving-average models in the 2-D adaptive parameter estimation problem, first developing a simple gradient algorithm then adding more sophistication.

There are two fundamental formulations of IIR adaptive filtering, commonly referred to as the equation error formulation (EEF) and the output error formulation (OEF). Equation error adaptive filters are similar to FIR filters in that they are characterized by a nonrecursive difference equation. They are two-input, single-output filters using the system input and the desired response as inputs. Since the structure is nonrecursive, the development of a gradient adaptive algorithm is straightforward. Unfortunately, they minimize an equation error and not an output error, generally resulting in a biased solution. However, EEF filters do have quadratic mean-squared error surfaces, which eliminates the concern of suboptimal local stationary points. The primary focus of this work is to investigate the development of 2-D recursive adaptive filtering with the output error formulation.

The output error filter is characterized by a recursive difference equation, which complicates the form of adaptive algorithms since, again, the error is now a nonlinear function of the coefficients. The objective of any adaptive algorithm is to minimize some performance criteria based on the prediction error. The most common cost function is the mean-squared error (MSE), $\xi=E[e^2]$, where e is the prediction error. As in the last chapter, we will approximate the MSE by the error squared in order to obtain a practical gradient estimate. The development of a 2-D IIR gradient-based adaptive algorithm begins by considering a direct-form structure in a 2-D system identification configuration as in Figure 3.1 where the unknown filter's observable response to the input signal $u(n_1,n_2)$ is

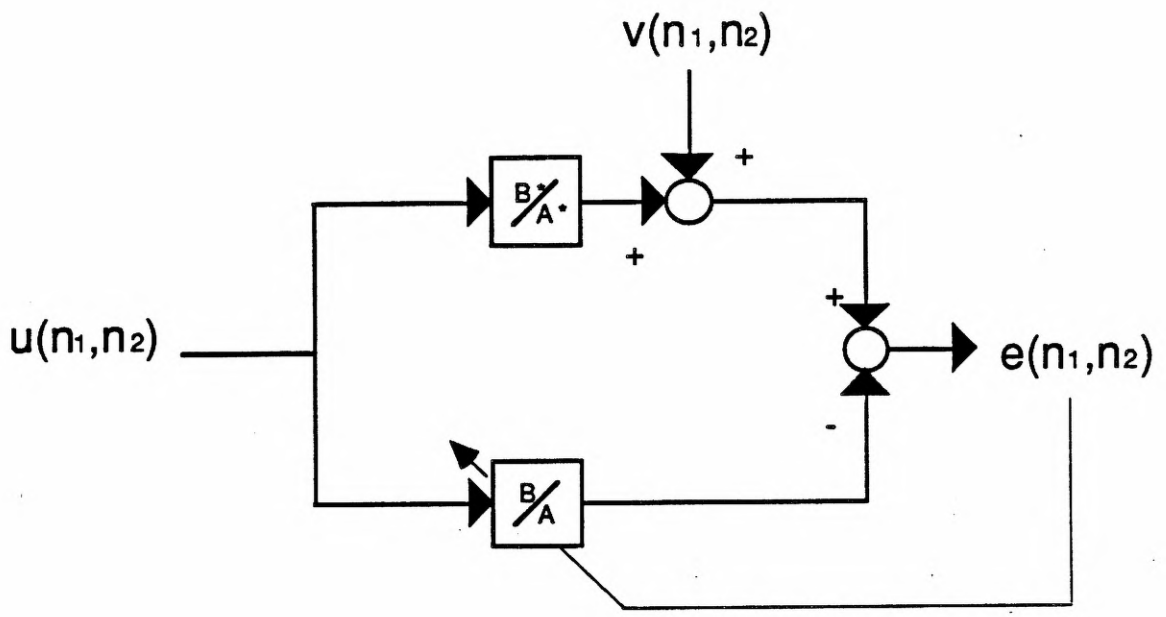


Figure 3.1 2-D system identification.

$$d(n_1,n_2) = \sum_{m_1=0}^{N_b^*} \sum_{m_2=0}^{N_b^*} u(n_1-m_1,n_2-m_2)b^*(m_1,m_2)$$

$$+ \sum_{m_1=0}^{N_a^*} \sum_{m_2=0}^{N_a^*} d(n_1-m_1,n_2-m_2)a^*(m_1,m_2) + v(n_1,n_2)$$

$$(m_1, m_2) \neq (0, 0) \qquad (3.1.a)$$

Taking the Z-transform of both sides of (3.1.a) gives the filter transfer function in terms of the fixed filter's coefficients

$$H^*(z_1,z_2) = \frac{B^*(z_1,z_2)}{A^*(z_1,z_2)} \qquad (3.1b)$$

For simplicity as well as stability concerns, we will consider only square first-quadrant quarter-plane filters with all of the singularities of the denominator coefficient array assumed to lie in the open unit bidisc defined as ($U^2 \triangleq \{(z_1,z_2): |z_1| < 1, |z_2| < 1\}$). We also assume that $H^*(z_1,z_2)$ is devoid of nonessential singularities of the second kind on the unit bidisc. The signal $v(n_1,n_2)$ is an additive uncorrelated noise source. Likewise, the adaptive filter's recursive difference equation is

$$y(n_1,n_2) = \sum_{m_1=0}^{N_b} \sum_{m_2=0}^{N_b} u(n_1-m_1,n_2-m_2)b(m_1,m_2)$$

$$+ \sum_{m_1=0}^{N_a} \sum_{m_2=0}^{N_a} y(n_1-m_1,n_2-m_2)a(m_1,m_2) \qquad (3.1.c)$$

$$(m_1, m_2)\neq(0, 0)$$

Considering the adaptive filter's coefficients fixed and unknown, we can also define a transfer function as

$$H(z_1,z_2) = \frac{B(z_1,z_2)}{A(z_1,z_2)} \qquad (3.1d)$$

Later, it will be necessary to include $a(0,0)$ in the adaptive parameter set. Doing so will introduce a distributed mean-squared error solution. The error signal, $e(n_1,n_2)$, is the difference between the desired response and the actual adaptive filter response. Using delay operator notation for the filters' coefficient arrays, i.e.,

$$A(q_1^{-1},q_2^{-1}) = 1 - \sum_{i=0}^{N_a} \sum_{j=0}^{N_a} a(i,j)q_1^{-i}q_2^{-j} \qquad (3.2.a)$$

$$(i,j)\neq(0,0)$$

$$B(q_1^{-1},q_2^{-1}) = \sum_{i=0}^{N_a} \sum_{j=0}^{N_a} b(i,j) q_1^{-i} q_2^{-j} \qquad (3.2.b)$$

Equations (3.1.a) and (3.1.c) can be rewritten as

$$d(n_1,n_2) = \frac{B^*(q_1^{-1},q_2^{-1})}{A^*(q_1^{-1},q_2^{-1})} u(n_1,n_2) + v(n_1,n_2) \qquad (3.3.a)$$

$$y(n_1,n_2) = \frac{B(q_1^{-1},q_2^{-1})}{A(q_1^{-1},q_2^{-1})} u(n_1,n_2) \qquad (3.3.b)$$

Furthermore, for notational convenience we shall arrange filter coefficients and signal arrays into column-ordered vectors. Define the information vector, $\phi_o(n_1,n_2)$, as

$$\phi_o(n_1,n_2) = \begin{bmatrix} y(n_1,n_2-1) \\ \vdots \\ y(n_1-N_a,n_2-N_a) \\ \hline u(n_1,n_2) \\ \vdots \\ u(n_1-N_b,n_2-N_b) \end{bmatrix} \qquad (3.4.a)$$

and the coefficient vector, $\theta_k(n_1,n_2)$, as

$$\theta_k(n_1,n_2) = \begin{bmatrix} a_k(0,1) \\ \vdots \\ a_k(N_a,N_a) \\ \hline b_k(0,0) \\ \vdots \\ b_k(N_b,N_b) \end{bmatrix} \tag{3.4.b}$$

where k represents a time-mapped index point within the 2-D raster, indicating the time dependence of θ. Notice that these vectors are constructed by concatenating columns of the 2-D coefficient arrays $a(n_1,n_2)$ and $b(n_1,n_2)$. The adaptive filter output can be expressed as

$$y(n_1,n_2) = \phi_o^t(n_1,n_2)\theta_k(n_1,n_2) \tag{3.5}$$

The mean squared error surface for 2-D IIR adaptive filters is not quadratic, but we can utilize a locally quadratic approximation to develop an acceleration algorithm. Traditional steepest descent algorithms are known to give slow convergence when used with IIR filters. We can use the quasi-Newton line search algorithm in this case by first assuming that we can approximate the cost function by a truncated Taylor series as follows

$$\xi(\theta) \cong \xi(\theta_k) + \nabla\xi(\theta_k)\,(\theta - \theta_k) + \frac{1}{2}\,(\theta - \theta_k)^T F(\theta_k)\,(\theta - \theta_k) \tag{3.6}$$

where $F(\theta_k)$ is the Hessian matrix. This function is minimized by Newton's algorithm

$$\theta_{k+1} = \theta_k - F(\theta_k)^{-1}\nabla\xi(\theta_k) \tag{3.7}$$

Now the Hessian matrix is the autocorrelation matrix $\boldsymbol{R}$. If ξ has continuous second partial derivatives near the optimal solution, then $\boldsymbol{R}$ is positive definite, and the method is well-defined with second-order convergence in that vicinity. To apply the quasi-Newton algorithm to our problem, we only have to include a convergence factor and recognize that we must use estimates of the autocorrelation matrix and the gradient vector. Now we can write the 2-D IIR quasi-Newton as follows

$$\theta_{k+1} = \theta_k - \frac{1}{2}\mu \hat{\boldsymbol{R}}_{\phi}^{-1} \hat{\nabla} \xi(\theta_k) \tag{3.8}$$

where μ is the convergence factor, $\hat{\boldsymbol{R}}_{\phi}^{-1}$ is the inverse information vector autocorrelation matrix estimate, and $\hat{\nabla}\xi(\theta_k)$ is the cost function gradient vector estimate. We have included the scaling factor in anticipation of using the LMS error approximation later. Equation (3.8) represents a sub-optimal quasi-Newton algorithm because the autocorrelation matrix estimate and the gradient vector estimate are used. Ideally, the update relation (3.8) would incorporate the true autocorrelation matrix $\boldsymbol{R}_{\phi}$ in order to compensate for the eigenvalue disparity introduced by a colored input signal, i.e.,

$$\boldsymbol{R}_{\phi} = \mathrm{E}\{\, \phi_o \, \phi_o^{t} \,\} = \mathrm{E}\begin{bmatrix} \boldsymbol{y}\boldsymbol{y}^{t} & \boldsymbol{y}\boldsymbol{u}^{t} \\ \boldsymbol{u}\boldsymbol{y}^{t} & \boldsymbol{u}\boldsymbol{u}^{t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{yy} & \boldsymbol{R}_{yu} \\ \boldsymbol{R}_{uy} & \boldsymbol{R}_{uu} \end{bmatrix} \tag{3.9}$$

Inclusion of (3.9), or an estimate of $\boldsymbol{R}_{\phi}$, reduces the effect of input statistics on the rate of convergence of the line search algorithm. The autocorrelation matrix must be positive semidefinite to ensure invertibility, and any subsequent method of estimating autocorrelation lags must take this into consideration.

Clearly the autocorrelation matrix $\boldsymbol{R}_{\phi}$ is time varying since ϕ_o is time varying. If we assume that the input signal is stationary, then $\boldsymbol{R}_{uu}$ in (3.9) is symmetric. Furthermore, $\boldsymbol{R}_{uu}$ is Toeplitz-block Toeplitz. An acceleration technique with low complexity requires the information vector autocorrelation matrix block components to be block Toeplitz. This special structure can be

exploited using either the block Levinson algorithm or the block preconditioned conjugate gradient algorithm to give a computationally efficient method for calculating the matrix-vector product required in the quasi-Newton recursion. The 2-D IIR adaptive filter must be subtly modified so that the adaptive parameter set includes the leading denominator coefficient, $a(0,0)$. Before including this coefficient in the coefficient vector, the 2-D autocorrelation matrix is dimension $[(N_a+1)^2-1+(N_b+1)^2]\times[(N_a+1)^2-1+(N_b+1)^2]$ and is not quite block Toeplitz. If the convergence factor is very small so that the filter coefficients change very slowly, then $\boldsymbol{R}_{yy}$, $\boldsymbol{R}_{yu}$ and $\boldsymbol{R}_{uy}$ can also be assumed to be Toeplitz-block Toeplitz.

In order to serve as a preconditioner in an acceleration algorithm, $\boldsymbol{R}_{\phi}$ will be estimated by ignoring the cross-correlation matrices. Without utilizing this assumption, algebraic manipulations exploiting a special matrix structure can not offer any computational savings. This will not introduce any bias into the solution, although performance will suffer. Toeplitz-block Toeplitz structure will be imposed on $\boldsymbol{R}_{yy}$ by assuming that, as above, the filter is slowly varying and $a(0,0)$ is included as an adaptive parameter. And finally, an appropriately biased autocorrelation lag estimate will be utilized to construct estimates of $\boldsymbol{R}_{yy}$ and $\boldsymbol{R}_{uu}$. The resulting information vector autocorrelation matrix estimate becomes

$$\hat{\boldsymbol{R}}_{\phi} = \begin{bmatrix} \hat{\boldsymbol{R}}_{yy} & 0 \\ 0 & \hat{\boldsymbol{R}}_{uu} \end{bmatrix} \tag{3.10}$$

Accordingly, the system $\hat{\boldsymbol{R}}_{\phi}\boldsymbol{x} = \hat{\nabla}_{\theta}(\xi)$ decouples into two Toeplitz-block Toeplitz systems enabling the use of several different fast numerical solution methods. Among the several important advantages of this method are faster convergence, numerical robustness, and better tracking performance. The incurred increase in computational complexity is modest and easily justified for low-order IIR filters. Further modifications remain necessary when designing a practical algorithm. The acceleration algorithm can be expressed more generally as

$$\theta_{k+1} = \theta_k - \mu \boldsymbol{M}_k \nabla \xi(\theta_k) \tag{3.11}$$

where $M_k\nabla\xi(\theta_k)$ is a search direction which is a) in the direction of steepest descent if $M_k=I$, or b) directly towards the optimal solution if $M_k=R_k^{-1}$. For any general point in the 2-D IIR coefficient space, the direction vector $M_k\nabla\xi(\theta_k)$ may not cause the value of $\xi(\theta_k)$ to decrease as μ increases from zero. This could result with a nonpositive definite M_k. Also, estimation noise at flat regions of the surface can cause erratic convergence behavior We can, however, use $M_k=[\varepsilon_k I+R_k]^{-1}$ to obtain good global and local convergence properties. The factor ε_k can be spatially varying as indicated by the subscript, allowing it to take large values on relatively "flat" regions of the error surface. Then ε_k can be adjusted towards zero as the coefficient vector approaches a region in which the quadratic approximation is more reasonable.

3.2 Derivation of Simplified Error Gradient Components

For simplicity, we will begin by crudely estimating the autocorrelation matrix as the identity matrix I. While providing no compensation for the statistical properties of the input, it serves as a basis for comparison. The 2-D IIR gradient components can be obtained just as in the 1-D IIR case. The adaptive algorithm should minimize the mean-squared error cost function. In order to descend the MSE performance surface, we resort to the ubiquitous LMS approximation

$$\xi = \mathrm{E}[e^2(n_1,n_2)] \approx e^2(n_1,n_2) \tag{3.12}$$

which allows for a simple and computationally efficient, albeit noisy, gradient estimate.

$$\begin{aligned}\hat{\nabla}_\theta(\xi) \approx \nabla_\theta\{e^2(n_1,n_2)\} &= -2\, e(n_1,n_2)\, \nabla_\theta(y(n_1,n_2)) \\ &= -2\, e(n_1,n_2)\, \nabla_\theta(\phi_o{}^t\theta)\end{aligned} \tag{3.13}$$

As the development proceeds, the expression for the gradient components becomes

$$\frac{\partial y(n_1,n_2)}{\partial a_k(m_1,m_2)} = y(n_1-m_1,n_2-m_2) + \sum_{l_1=0}^{N_a} \sum_{l_2=0}^{N_a} a_k(l_1,l_2) \frac{\partial y(n_1-l_1,n_2-l_2)}{\partial a_k(m_1,m_2)}$$

$$(l_1,l_2)\neq(0,0)$$

$$\frac{\partial y(n_1,n_2)}{\partial b_k(m_1,m_2)} = u(n_1-m_1,n_2-m_2) + \sum_{l_1=0}^{N_a} \sum_{l_2=0}^{N_a} a_k(l_1,l_2) \frac{\partial y(n_1-l_1,n_2-l_2)}{\partial b_k(m_1,m_2)}$$

$$(l_1,l_2)\neq(0,0) \qquad (3.14)$$

where k indicates coefficients at the image spatial index (n_1,n_2). The presence of the summation terms in (3.14) is required since past output samples are related to present filter parameters through the adaptive algorithm. Since the derivatives on the right-hand side of (3.14) are with respect to a_k and b_k, the expression is not recursively computable. Utilizing a small μ approximation so that we may assume the coefficients are slowly varying, it is possible to approximate (3.14)

$$\frac{\partial y(n_1,n_2)}{\partial a_k(m_1,m_2)} = y(n_1-m_1,n_2-m_2) + \sum_{l_1=0}^{N_a} \sum_{l_2=0}^{N_a} a_k(l_1,l_2) \frac{\partial y(n_1-l_1,n_2-l_2)}{\partial a_{k-l}(m_1,m_2)}$$

$$(l_1,l_2)\neq(0,0)$$

$$\frac{\partial y(n_1,n_2)}{\partial b_k(m_1,m_2)} = u(n_1-m_1,n_2-m_2) + \sum_{l_1=0}^{N_a} \sum_{l_2=0}^{N_a} a_k(l_1,l_2) \frac{\partial y(n_1-l_1,n_2-l_2)}{\partial b_{k-l}(m_1,m_2)}$$

$$(l_1,l_2)\neq(0,0) \qquad (3.15)$$

where the $(k-l)$ subscript indicates the coefficients at the (n_1-l_1,n_2-l_2) spatial iteration point. Equation (3.15) may then be rewritten using delay operator notation

$$\nabla_{a(m1,m2)}(n_1,n_2) = \frac{\partial y(n_1,n_2)}{\partial a_k(m_1,m_2)} = \frac{1}{A(n_1,n_2,q_1^{-1},q_2^{-1})} y(n_1-m_1,n_2-m_2)$$

$$\nabla_{b(m1,m2)}(n_1,n_2) = \frac{\partial y(n_1,n_2)}{\partial b_k(m_1,m_2)} = \frac{1}{A(n_1,n_2,q_1^{-1},q_2^{-1})}\, u(n_1-m_1,n_2-m_2) \tag{3.16}$$

Therefore, the gradient components in (3.16) are generated using $(N_b+1)^2+(N_a+1)^2-1$ identical all-pole filters in parallel, with each input being simply delayed versions of the adaptive filter input and output. The implementation of (3.16) imposes significant computational and storage requirements on the algorithm. If we then assume that $A(n_1-m_1,n_2-m_2,q_1^{-1},q_2^{-1}) = A(n_1,n_2,q_1^{-1},q_2^{-1})$, (3.16) becomes

$$\begin{aligned}\nabla_{a(m1,m2)}(n_1,n_2) &= \frac{y(n_1-m_1,n_2-m_2)}{A(n_1-m_1,n_2-m_2,q_1^{-1},q_2^{-1})} \\ &= \nabla_{a(0,0)}(n_1-m_1,n_2-m_2) \\ \nabla_{b(m1,m2)}(n_1,n_2) &= \frac{u(n_1-m_1,n_2-m_2)}{A(n_1-m_1,n_2-m_2,q_1^{-1},q_2^{-1})} \\ &= \nabla_{b(0,0)}(n_1-m_1,n_2-m_2)\end{aligned} \tag{3.17}$$

Equation (3.17) gives the gradient estimate of the output with respect to each filter coefficient using much less complexity than would be required without the assumption. The validity of the approximations involved in deriving (3.17) depends on the application and the geometry of the raster indexing. Obviously long horizontal scanning with retrace may invalidate the assumption that $A(n_1,n_2,q_1^{-1},q_2^{-1}) = A(n_1-m_1,n_2-m_2,q_1^{-1},q_2^{-1})$. Although $A(n_1,n_2,q_1^{-1},q_2^{-1})$ may still closely approximate $A(n_1-m_1,n_2,q_1^{-1},q_2^{-1})$, corresponding to the horizontal n_1 direction, an alternative approach would be to use a diagonal indexing scheme, which may optimize performance under these assumptions by minimizing the absolute delay between all index points within any given filter mask.

Equation (3.17) provides an efficient means of calculating gradient vector estimates, which is now $O[N_a^2]$ instead of $O[N_a^4]$. The gradient components in (3.16) differ only in that the appropriate input signals are delayed versions of each other. The simplified version in (3.17) results since filtering with a slowly time-varying filter will produce outputs which are approximately shifted versions of each other. The savings in storage requirements and computational complexity are significant, especially for the 2-D filters under consideration. This approximation should not severely degrade performance. If wide horizontal scan lines do ultimately interfere with convergence, it is possible to step backwards in the development of the gradient terms and assume spatial invariance only in the major scanning direction.

3.3 Summary of Two-Dimensional IIR Adaptive Filtering Algorithms

With $A(n_1,n_2,q_1^{-1},q_2^{-1}) = 1$ and $\boldsymbol{R}_\phi = \boldsymbol{I}$, the quasi-Newton algorithm reduces to the LMS algorithm for 2-D FIR filters. The 2-D IIR quasi-Newton algorithm in its entirety is

2-D IIRQN:
$$\theta_{k+1} = \theta_k + \mu \hat{R}_\phi^{-1} \nabla_\theta\{y(n_1,n_2)\}\, e(n_1,n_2) \tag{3.18a}$$

The LMS follows directly as described above.

2-D IIRLMS:
$$\theta_{k+1} = \theta_k + \mu \nabla_\theta\{y(n_1,n_2)\}\, e(n_1,n_2) \tag{3.18b}$$

with the gradient components specified in the previous section. Again, for horizontal scanning the vector indices are as follows

$$k = (n_1,n_2) \qquad k+1 = (n_1+1,n_2) \tag{3.19}$$

except during retrace. The structure required to compute the gradient components is shown in Figure 3.2.

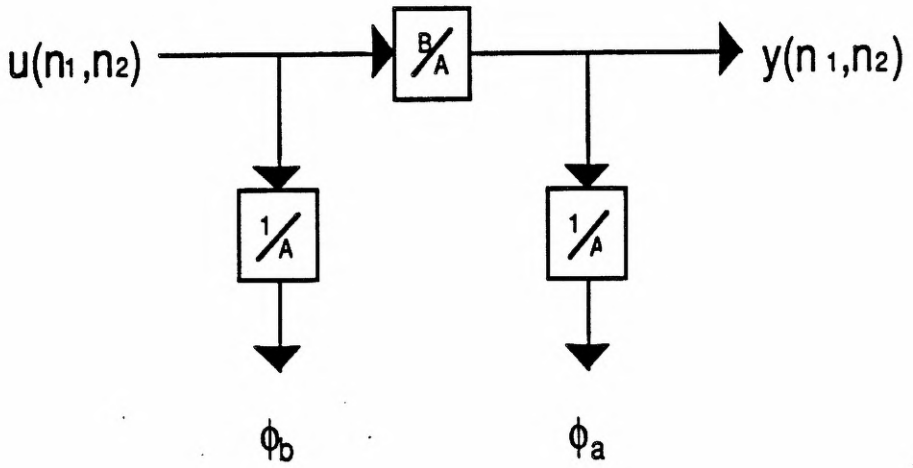


Figure 3.2 IIR gradient components.

3.4 Two-Dimensional IIR LMS Experiments

Computer experiments with system identification show the very unique behavior of 2-D IIR adaptive filters. We will first discuss experiments using the LMS algorithm in (3.18) with $\boldsymbol{R}_{\phi} = \boldsymbol{I}$, meaning the autocorrelation matrix is crudely estimated to be the identity matrix. In Section 3.4 we will include an estimate of $\boldsymbol{R}_{\phi}$ so that we can observe the possible improvements in rate of convergence. With the configuration shown in Figure 3.1, our computer experiments use an input process which is zero-mean, unit-variance white noise. The next section will address cases in which the input signal is colored. The convergence plots which follow show the absolute value of the system error in decibels vs. the iteration number, where the time-mapped iteration is dependent upon the geometry of the indexing scheme. In Figure 3.3, we show four of the possible indexing methods available. The type of indexing scheme used is obviously limited due to computability issues. For example, currently we are restricting the region of support of the IIR denominator polynomial to be in the lower-left quarter plane referenced to the current index point (n_1,n_2).

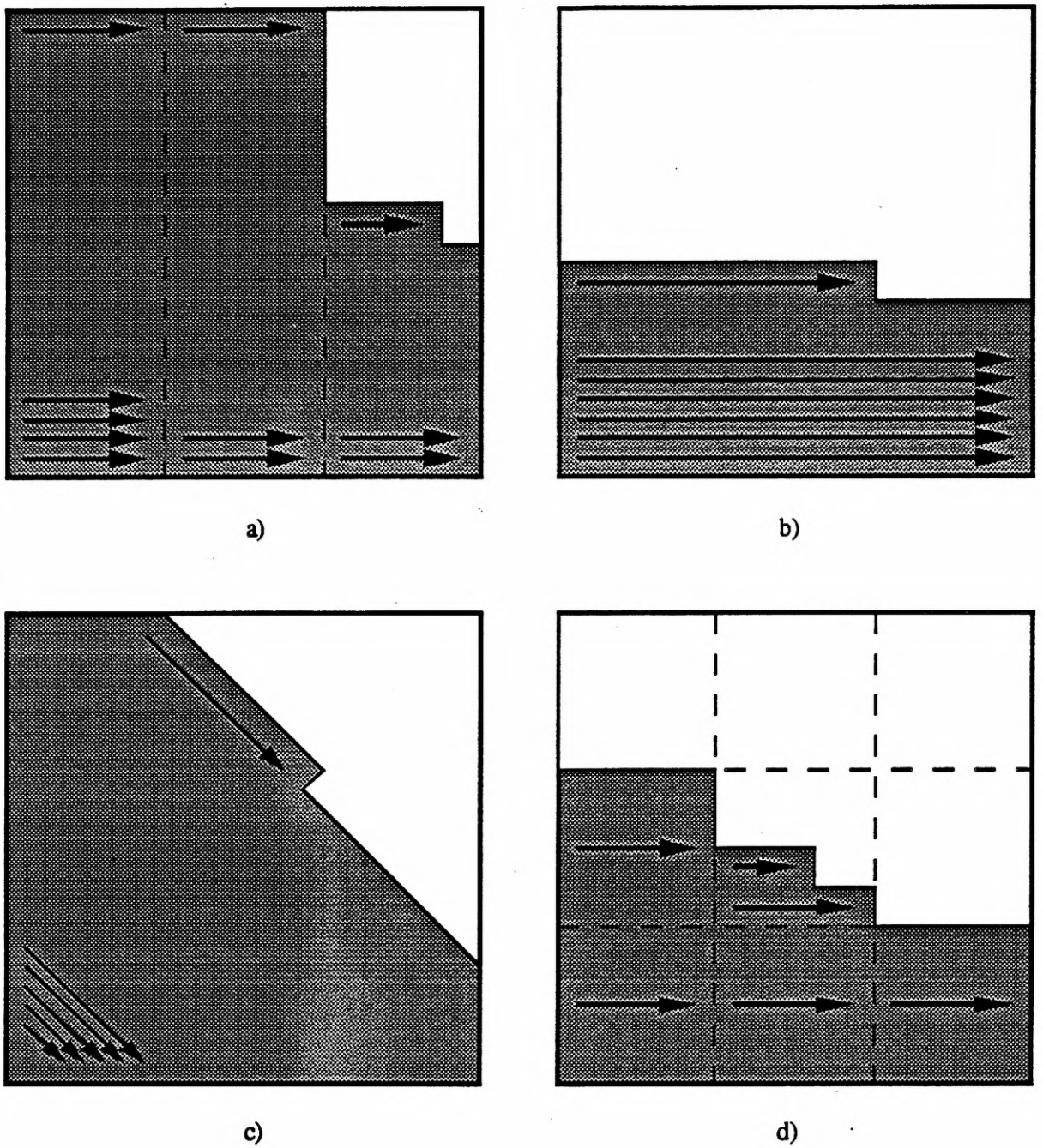


Figure 3.3 Indexing schemes - a) rectangular indexing with short scan lines, b) rectangular indexing with image-length scan lines, c) diagonal indexing, and d) block diagonal indexing.

Therefore, to compute an output sample at the current spatial index, we need previously computed outputs at points to the left of and beneath the current index, including on the current row and column. So, combinations of rectangular and diagonal indexing are certainly applicable, as shown below.

First, the PxQ image can be segmented into vertical strips of width L. Each segment can be processed consecutively using rectangular indexing. In Figure 3.3b) the image is scanned using a common horizontal raster running the width of the image. Diagonal indexing is shown in Figure 3.3c), and this may have certain advantages for the work being presented. Finally, a combination of these can be obtained by segmenting the image into square blocks and processing them on a diagonal pattern. Each block can then be processed using rectangular or diagonal indexing as required.

We discuss different indexing patterns because 2-D IIR adaptive filtering depends on the indexing scheme used to process the data. There are a couple of reasons for this phenomenon. First, 2-D IIR adaptive filters are nonlinear and shift variant. Second, the approximations utilized in deriving the gradient estimates in the last section have varying degrees of validity for different indexing schemes. Obviously, in order for the slowly varying coefficient assumption to be valid, we want the scan lines to be as short as possible so that the data within the denominator filter mask would have been computed with coefficients close in value to those which are current. Stated another way, we want the "age" of the coefficients used to compute those samples to be small. Now, using diagonal indexing, the scan lines begin with a length of one and increase by one after indexing to the next consecutive diagonal line. Using the scheme shown in Figure 3.3a), the lines are always of length L, and since L is smaller than the image width, we expect better performance within a given image segment. However, when processing the next segment, "old" data are present along the boundary of the two so that performance is affected.

Figures 3.4, 3.5, and 3.6 show time-mapped convergence plots for the same second-order adaptive filter in system identification. The input is a 100x100 white noise array, and the adaptive filter is matched order, meaning the unknown filter and the adaptive filter are both of the same

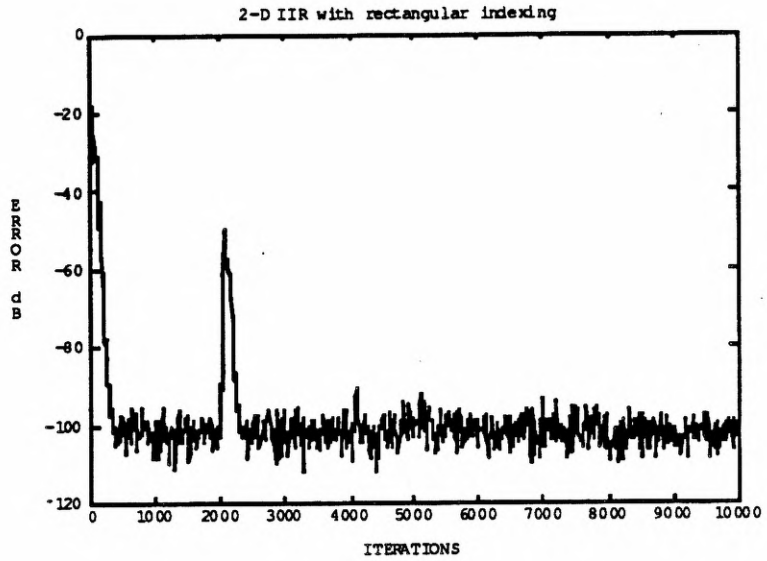


Figure 3.4 Convergence plot for a 2-D, second-order, (AR) IIR LMS adaptive filter using rectangular indexing with l=20.

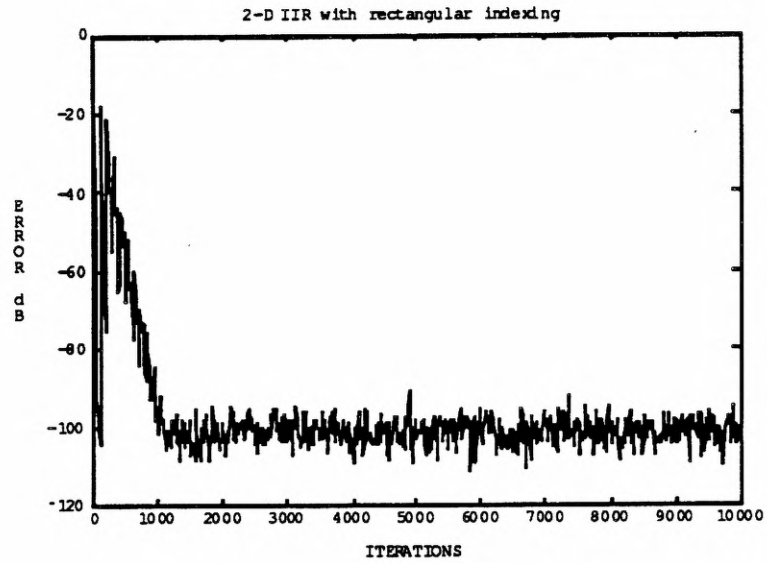


Figure 3.5 Convergence plot for a 2-D, second-order, (AR) IIR LMS adaptive filter using rectangular indexing with l=100.

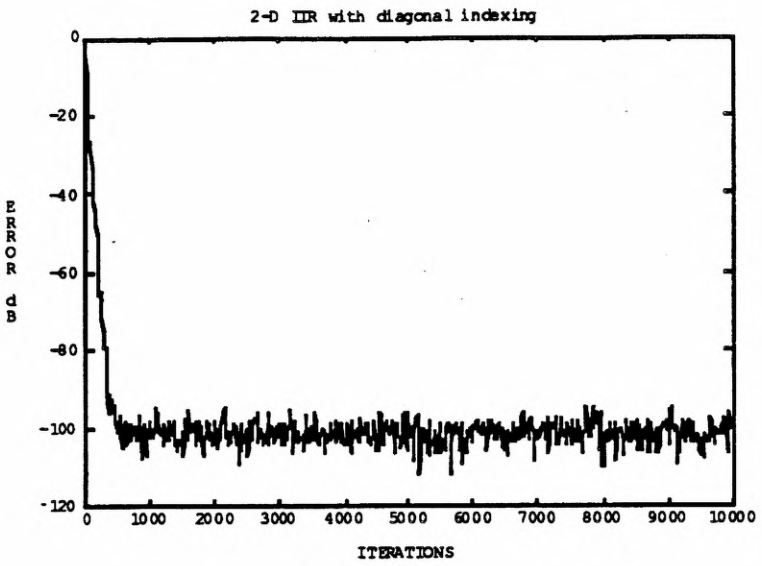


Figure 3.6 Convergence plot for a 2-D, second-order, (AR) IIR LMS adaptive filter using diagonal indexing.

order. The numerator in this case is a constant, so we will refer to the structure as autoregressive (AR). Autoregressive-moving average (ARMA) will refer to filters with N_a and N_b greater than zero. Figure 3.4 shows the convergence plot using the rectangular indexing scheme in Figure 3.3a) with a scan line width L=20. Therefore, the image is segmented into five 20x100 strips. Clearly, at iteration number 2001 some reinitialization occurs as we index from the first segment to the second. However, within each segment convergence is rapid, and at the beginning of the third segment, this reinitialization is barely evident around the -100 dB noise floor. Figure 3.5 shows the convergence results using the indexing pattern in Figure 3.3b), which is a simple raster scan. The filter converges as the index point approaches the edge of the image within each scan line, only to be partially reinitialized upon retrace. Clearly, convergence is directionally dependent and must occur in both directions. Figures 3.6 and 3.7 show the convergence plot using diagonal indexing as in Figure 3.3c). Figure 3.7 is an expanded view of the first 400 iterations of Figure 3.6. Evident from these two plots is the fact that convergence is very rapid at the beginning with very short scan lines. As the scan lines grow from a length of one to a maximum of 100, convergence gradually slows, giving the learning curve a concave shape. Not shown here because of the

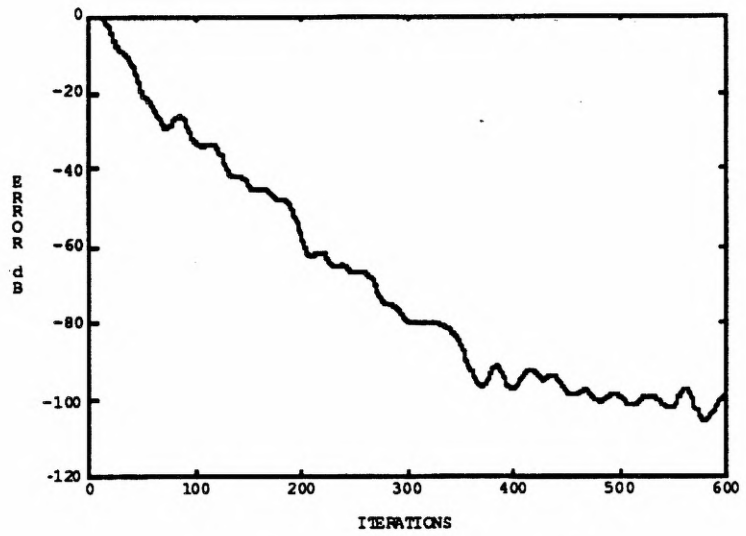


Figure 3.7 Convergence plot from Figure 3.6 expanded to show the reduction in the rate of convergence for diagonal indexing as the scan lines get increasingly longer.

circumstances, if the filter were not converged after reaching the main diagonal of the image, the rate of convergence would begin increasing once past the main diagonal since the scan lines subsequently decrease after that. Usually in real applications we are dealing with nonstationary data or slowly varying systems so that overall performance, and not just absolute rate of convergence, is a concern.

Figures 3.8, 3.9, and 3.10 show the convergence plots from Figures 3.4, 3.5, and 3.6, consecutively, in mesh form so as to better illustrate the relationship between index geometry and rate of convergence. In each case, the origin is labeled (0,0), and the x-direction is parallel to the page going from right to left.

Figures 3.11, 3.12, and 3.13 show time-mapped convergence plots for a third-order AR filter under the same circumstances. These three experiments show the reduction in rate of convergence as the filter order increases. Some of the characteristics associated with the different indexing methods are more pronounced here with the slower convergence. The saw-tooth shape is very clear from the learning curve as partial reinitialization occurs. A first-order ARMA filter is used to generate the learning curve in Figure 3.14. Note that including a nontrivial numerator

polynomial also adversely affects convergence. It takes this structure about 1500 iterations to reach the noise floor. Also note that now we concentrate on absolute rate of convergence, so the results are presented using the indexing scheme in Figure 3.3a). However, after the filter has converged, we discontinue adaptation and do not process the next segment. A second-order ARMA filter produced the learning curve in Figure 3.15. Performance rapidly deteriorates as the order of the ARMA structure is increased. This structure requires 16000 iterations to reach the -100 dB noise floor. It appears that the performance problems observed for the higher-order IIR filters are due to a combination of increased complexity and nonlinearity inherent to adding IIR coefficients and also to the effect of ARMA coloring of the information vector autocorrelation matrix. Even if the input process is white, the adaptive filter colors the data so that R_{yy}, R_{uy}, and R_{yu} are not diagonal. This spreads the eigenvalues, thus reducing the rate of convergence.

Finally, Figures 3.16 and 3.17 show several learning curves for a first-order AR filter structure given as

$$H(z_1,z_2) = \frac{1}{(1+p_1 z_1^{-1})(1+p_2 z_2^{-1})} \tag{3.20}$$

Since the structure is separable, we can place the singularities p_1 and p_2 at our discretion. In Figure 3.16 we show the learning curves for the cases of p_1=p_2=0.1, 0.3, and 0.5 on the same plot. With $p_{1,2}$=0.1 the filter reaches the noise floor in about 200 iterations. As the poles are moved out to 0.5, the rate of learning slows so that it takes about 1200 iterations to converge. From Figure 3.17, with the poles at 0.7 and 0.8, the rate of convergence is severely hampered so that, for the first case, it takes 8000 iterations to reach -80 dB.

Throughout this LMS experimentation, it appears that 2-D IIR error surface characteristics may be very similar to those of 1-D IIR adaptive filters with respect to conditions for the existence of unique or multiple minima.

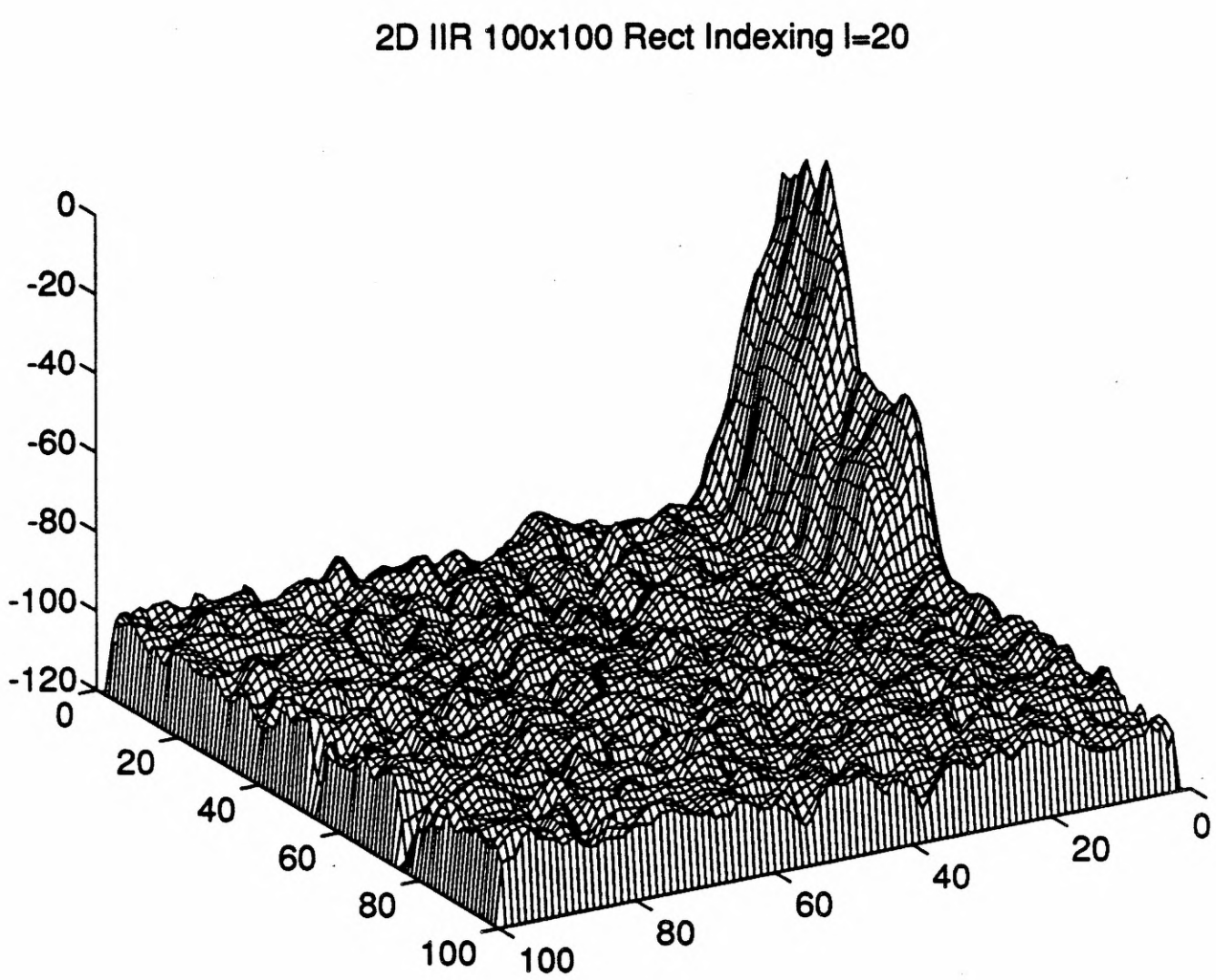


Figure 3.8 Convergence plot in mesh form for the 2-D, second-order, (AR) IIR adaptive filter using rectangular indexing with l=20 from Figure 3.4.

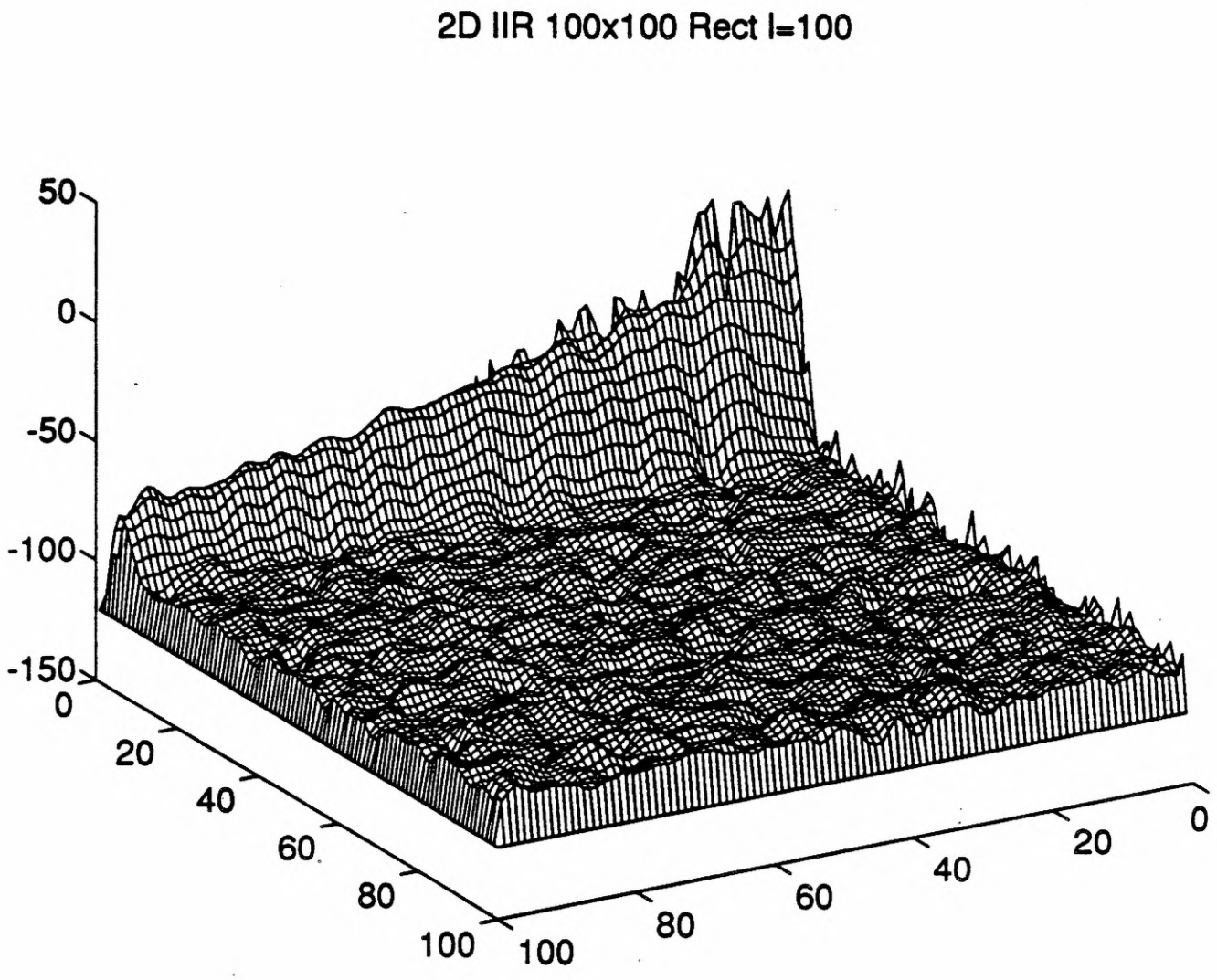


Figure 3.9 Convergence plot in mesh form for the 2-D, second-order, (AR) IIR adaptive filter using rectangular indexing with l=100 from Figure 3.5.

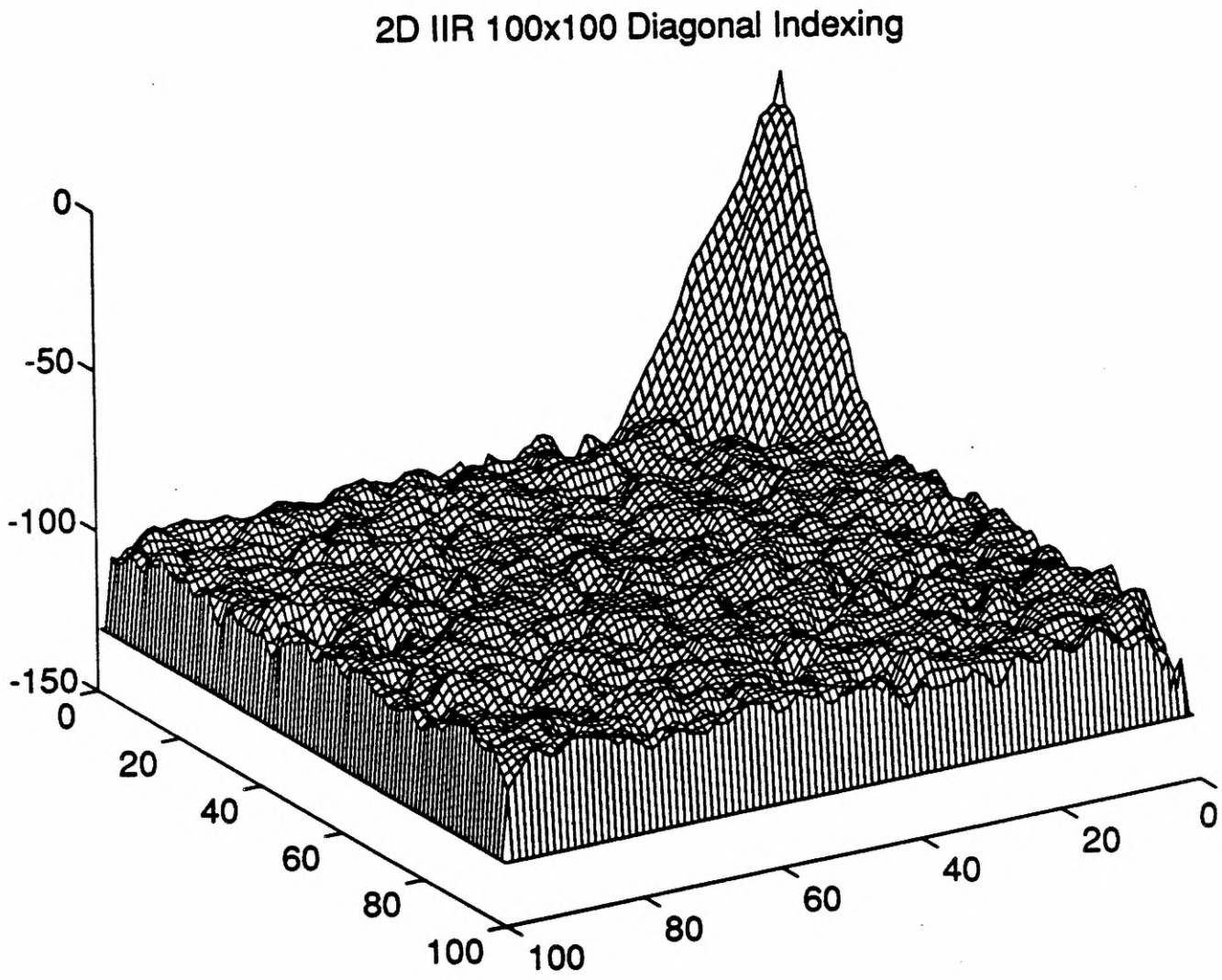


Figure 3.10 Convergence plot in mesh form for the 2-D, second-order, (AR) IIR adaptive filter using diagonal indexing from Figure 3.6.

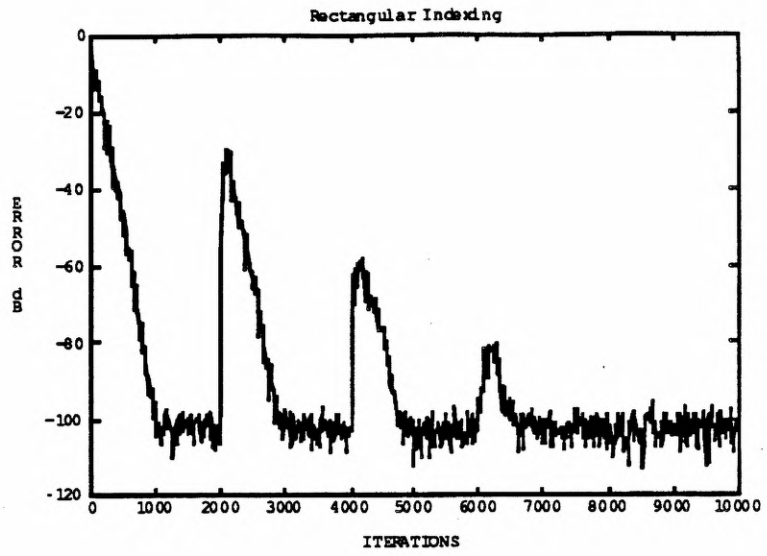


Figure 3.11 Convergence plot for a 2-D, third-order, (AR) IIR LMS adaptive filter using rectangular indexing with l=20.

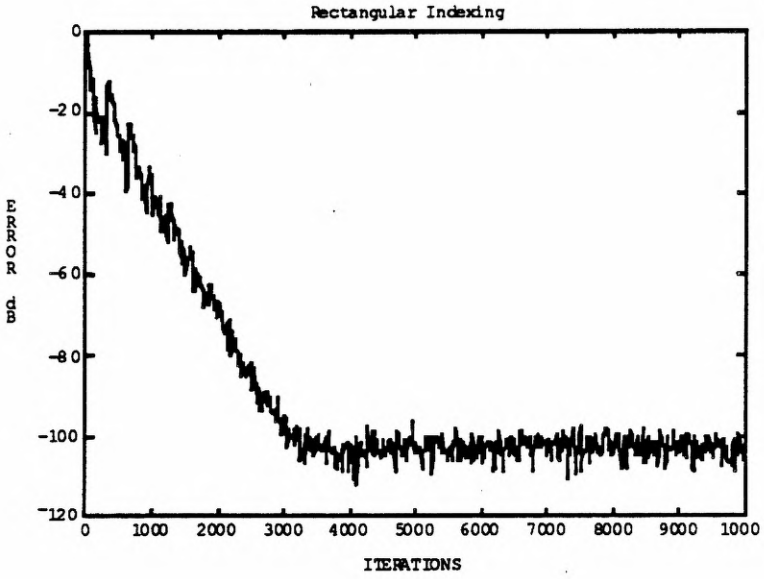


Figure 3.12 Convergence plot for a 2-D, third-order, (AR) IIR LMS adaptive filter using rectangular indexing with l=100.

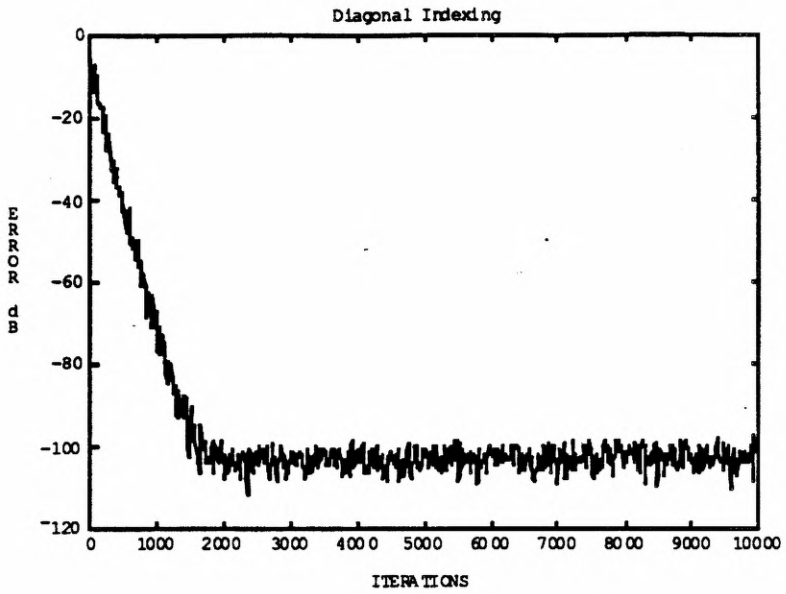


Figure 3.13 Convergence plot for a 2-D, third-order, (AR) IIR LMS adaptive filter using diagonal indexing.

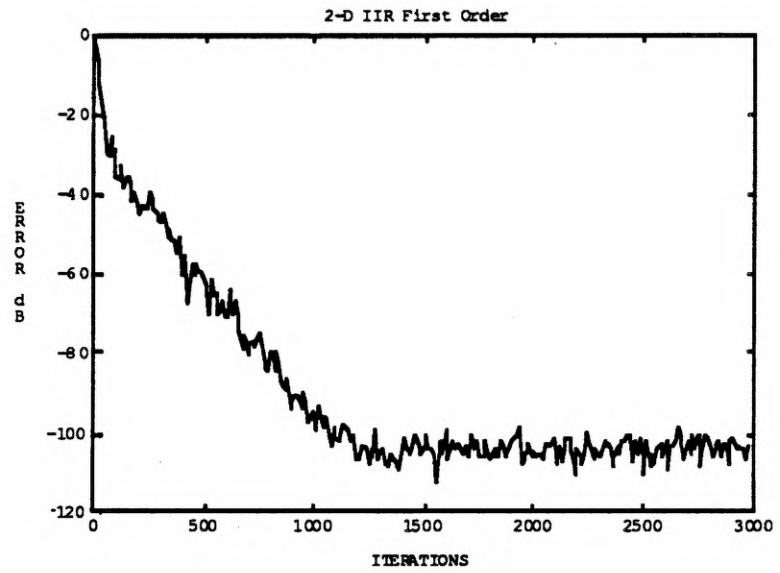


Figure 3.14 Convergence plot for a 2-D, first-order, (ARMA) IIR LMS adaptive filter using rectangular indexing with l=20.

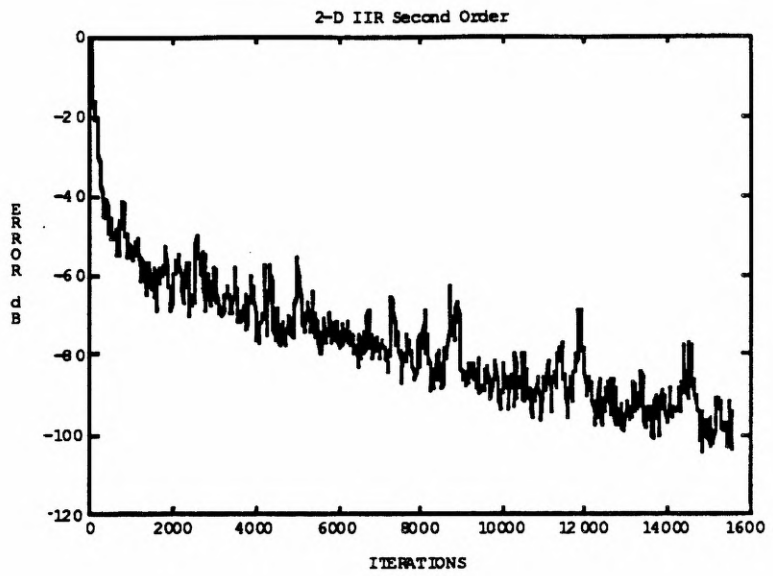


Figure 3.15 Convergence plot for a 2-D, second-order, (ARMA) IIR LMS adaptive filter using rectangular indexing with l=20.

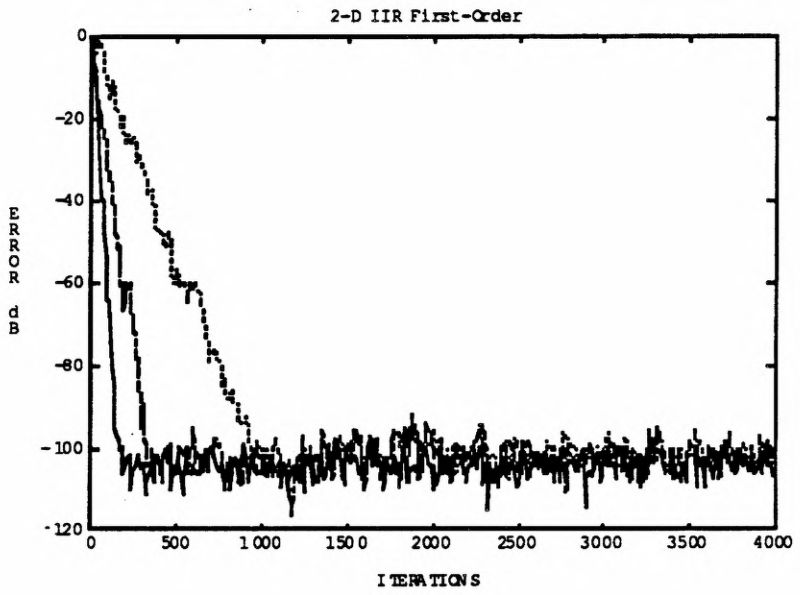


Figure 3.16 Convergence plot for a 2-D, first-order, separable (AR) IIR LMS adaptive filter with poles at 0.1 (lower), 0.3 (middle), and 0.5 (upper).

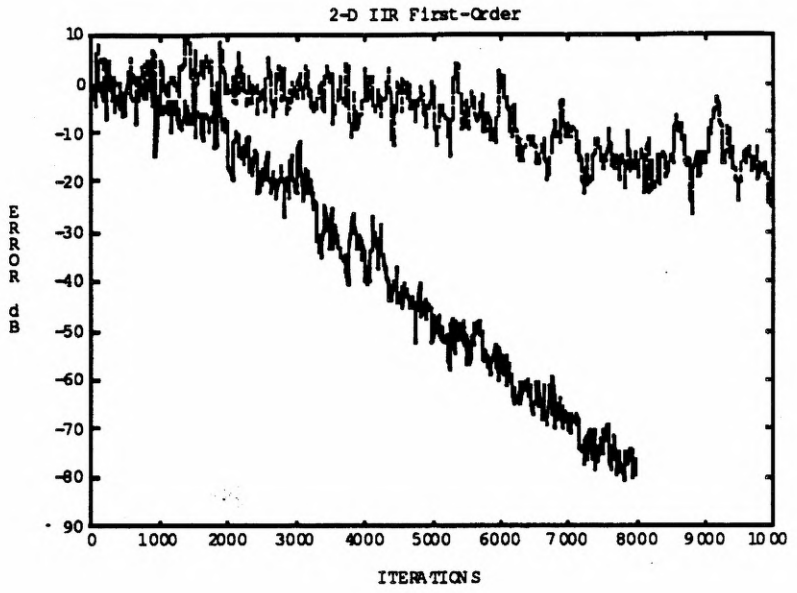


Figure 3.17 Convergence plot for the same 2D IIR LMS adaptive filter above with poles at 0.7 (lower) and 0.8 (upper).

3.5 Two-Dimensional IIR Quasi-Newton Experiments

The system identification experiments that follow utilize the 2-D FIR fast quasi-Newton algorithm from Chapter 2 with the 2-D IIR structure. Using the autocorrelation matrix estimate in (3.10), the IIR gradient solution vector is equivalent to two separate FIR systems, allowing efficient computation of the required gradient vector component. Also, the same autocorrelation lag estimate from Chapter 2 is used. The initial conditions must be set so that the coefficient vector is reasonably close to the region of the mean-squared error surface which can be approximated to be quadratic. This is a well-known limitation of the quasi-Newton algorithm regardless of the application under consideration.

For simplicity we begin with some autoregressive models, i.e., the numerator polynomial is a constant. In Figure 3.18, a second-order (AR) adaptive filter identifies a second-order low-pass model. For comparison the experiment was also executed using the LMS algorithm with both a white and a low-pass colored input signal. With white noise the LMS filter converged in about

300 iterations. Convergence of the LMS filter is then shown to be much slower with a colored input. Finally, the fast quasi-Newton filter is shown to dramatically accelerate convergence in the presence of colored noise. It reached the noise floor in approximately the same number of iterations as the white noise LMS filter. It should also be mentioned here that the filter being identified has relatively mild coloring characteristics of its own. It was shown in the last section how the location of the unknown filter's singularities affects the rate of convergence of IIR filters.

In Figure 3.19 we show the convergence plots for the separable, first-order, all-pole model filter from the last section using both the LMS filter and the FQN filter. The poles are located at 0.7 in both the z_1 and z_2 planes, and the input is white noise. The lower curve corresponds to the FQN filter, and it reaches the noise floor in less than 2000 iterations. The upper curve is the LMS filter under the same conditions. Clearly, the FQN filter outperforms the LMS filter even in the presence of white noise. The FQN algorithm is now compensating for the coloring effects of the filter itself upon the rate of convergence. The 2-D IIR autocorrelation matrix has disparate eigenvalues even if the input is white, since three of the four block elements of $\boldsymbol{R}$ ($\boldsymbol{R}_{yy}$, $\boldsymbol{R}_{uy}$, and $\boldsymbol{R}_{yu}$) are constructed with filtered versions of the input. Next we continue with the same separable all-pole filter, but we move the poles closer towards the distinguished boundary of the unit bidisc ($T^2 \triangleq \{(z_1,z_2): |z_1| = 1, |z_2| = 1\}$). With each pole at 0.8, the learning curves in Figure 3.20 show convergence slowing under the same circumstances. The FQN adaptive filter required 8000 iterations to converge to -100 dB, whereas the LMS filter now struggles to get down to -20 dB and continues to converge very slowly.

Autoregressive moving-average filters with both the numerator and denominator polynomials being second degree are used in Figure 3.21. Both curves use the same matched order filter with a white input. The lower curve shows the FQN filter converging in about 3000 iterations, and the upper curve shows the LMS filter at about -50 dB and continuing to converge slowly. In this case, since the input is white the FNQ filter uses the LMS algorithm to update the numerator coefficients while simultaneously using the FQN algorithm for the denominator coefficient

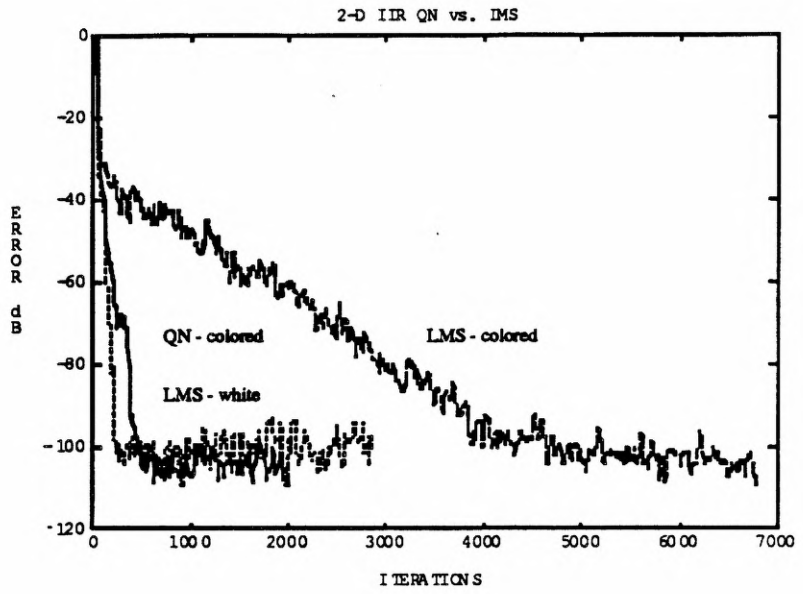


Figure 3.18 Convergence plot for a 2-D, second-order, (AR) IIR adaptive filter showing acceleration with the quasi-Newton algorithm in a colored signal environment.

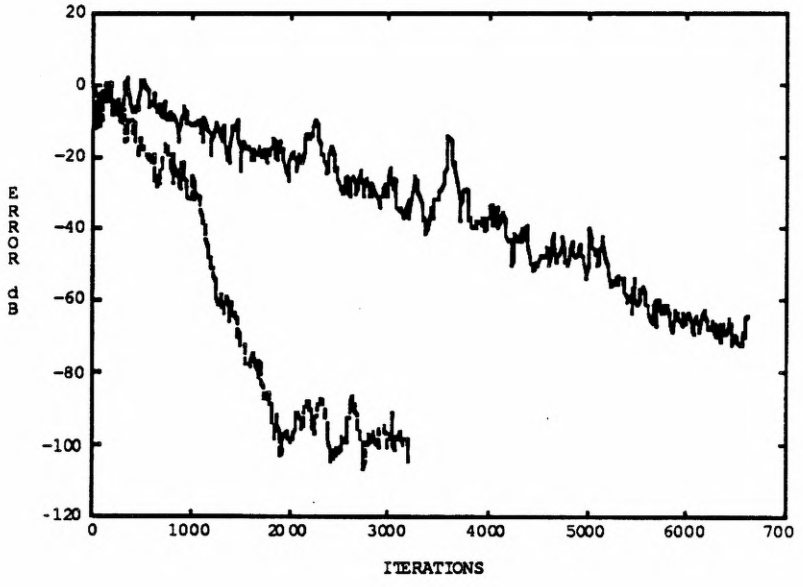


Figure 3.19 Convergence plot for a 2-D, first-order, separable (AR) IIR adaptive filter with poles at 0.7, comparing the fast quasi-Newton algorithm (lower) to the LMS algorithm (upper).

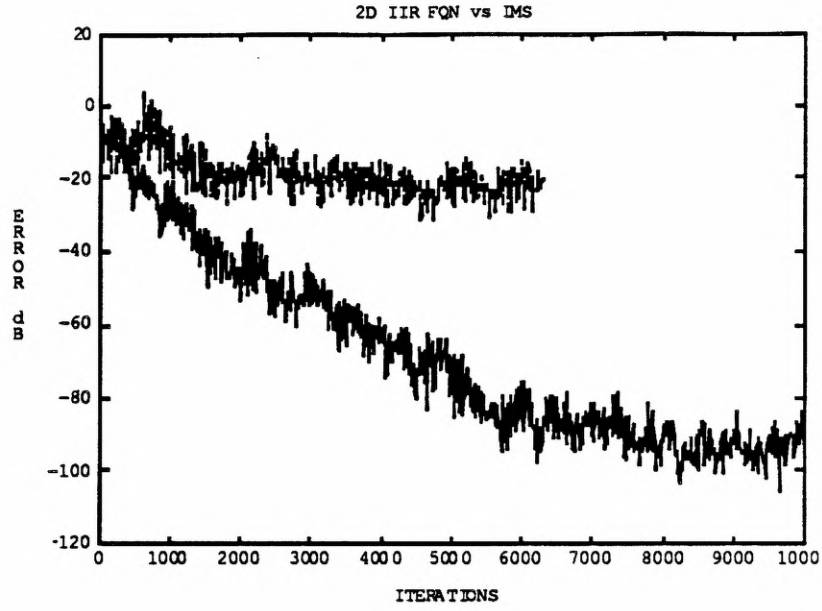


Figure 3.20 Convergence plot for a 2-D, first-order, separable (AR) IIR adaptive filter with poles at 0.8, comparing the fast quasi-Newton algorithm (lower) to the LMS algorithm (upper).

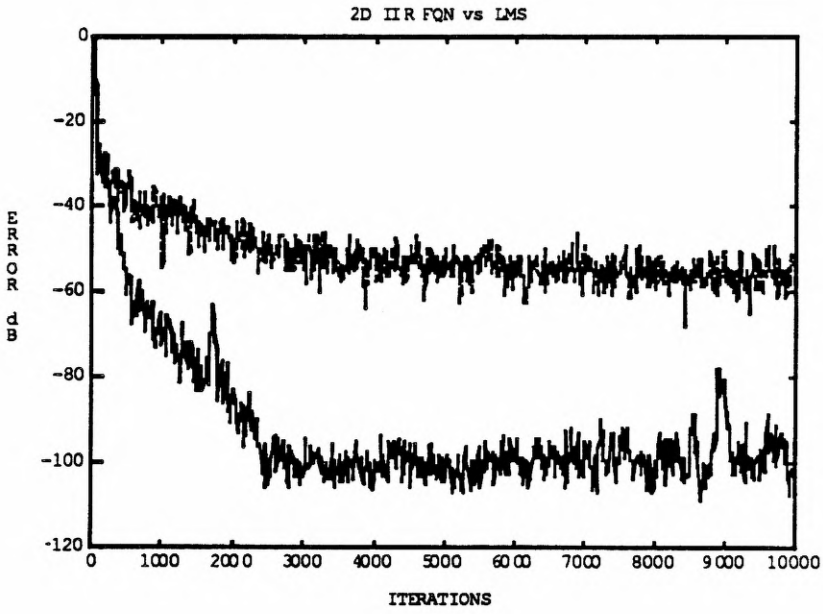


Figure 3.21 Convergence plot for a 2-D, second-order, (ARMA) IIR adaptive filter, comparing the fast quasi-Newton algorithm (lower) to the LMS algorithm (upper). The filter is second order in both the numerator and denominator.

3.6 Uniqueness Characteristics of the 2-D IIR MSE Minimization

Optimization problems involving recursive filters are complicated by the fact that the error measure is a nonlinear function of the coefficient vector. It is well-known that FIR filters produce a quadratic mean-squared error surface with a unique minimum solution. Gradient-based algorithms will descend the FIR error surface towards the global minimum as long as the input is persistently exciting. However, a gradient-based IIR algorithm is not guaranteed to reach a unique solution because of the possible existence of multiple minima. With the existence of local minima the filter can get stuck at a suboptimal solution. For 1-D IIR filters the existence and multiplicity of local minimum points are highly dependent on the structure of the IIR filter and the characteristics of the input [43]-[44].

In 1981, Stearns [40] conjectured that sufficient order IIR adaptive filters in system identification with white input produced unimodal error surfaces with the global mean-squared error equal to zero. Fan and Nayeri [44] presented proofs and some counterexamples in which they proved its validity for first- and second-order 1-D IIR adaptive filters. However, while second-order all-pole filters can be shown to be unimodal for matched order filters, overparameterization in this case can result in a multimodal error surface. They showed that the conjecture breaks down for higher-order filters unless a special relationship exists (Soderstrom and Stoica [41]) between the order of the adaptive filter's numerator and the order of the unknown filter's denominator. Specifically, if the number of free parameters in the adaptive filter's numerator is at least as great as the number of poles in the unknown filter, then Stearns' conjecture holds. This result can be modified and extended to the colored noise case, but it requires exact knowledge of the parameters of the autoregressive moving-average input process. This is a sufficient, but not necessary, condition to guarantee the unimodality of the 1-D error surface for system identification only. Nayeri [43] also showed that the existence of degenerated solutions for all-pole adaptive filters implies multimodality of the 1-D IIR error surface.

The usefulness of 2-D IIR adaptive filters similarly requires an investigation into the stationary point properties of the 2-D IIR error surface. Experimental evidence seems to indicate that 2-D IIR filters have error surface uniqueness properties very similar to those of 1-D IIR adaptive filters. No suboptimal solutions have been encountered with sufficient (first- and second-) order filters in system identification driven with white noise. Higher-order filters converge so slowly that uniqueness properties are difficult to ascertain experimentally, but their usefulness is severely limited anyway.

The analysis of the 2-D mean-squared error surface begins by considering the filter coefficients fixed but unknown, and then expanding the expression for the MSE from Equation (3.12)

$$E[e^2(n_1,n_2)] = E\left[\left\{\left(\frac{B^*}{A^*} - \frac{B}{A}\right)(q_1^{-1},q_2^{-1})\, u(n_1,n_2)\right\}^2\right] + E[v^2(n_1,n_2)] \tag{3.21}$$

which is, of course, a function of the coefficient vector. The stationary points of the 2-D MSE functional are found as the solution of the following equations obtained by equating the gradient of (3.21) to zero.

$$E\left[\left(\frac{B^*(q_1^{-1},q_2^{-1})}{A^*(q_1^{-1},q_2^{-1})} - \frac{B(q_1^{-1},q_2^{-1})}{A(q_1^{-1},q_2^{-1})}\right)u(n_1,n_2)\ \frac{1}{A(q_1^{-1},q_2^{-1})}\ u(n_1-r_1,n_2-r_2)\right] = 0$$

$$0 \le r_1,r_2 \le N_b$$

$$E\left[\left(\frac{B^*(q_1^{-1},q_2^{-1})}{A^*(q_1^{-1},q_2^{-1})} - \frac{B(q_1^{-1},q_2^{-1})}{A(q_1^{-1},q_2^{-1})}\right)u(n_1,n_2)\ \frac{B(q_1^{-1},q_2^{-1})}{A^2(q_1^{-1},q_2^{-1})}\ u(n_1-r_1,n_2-r_2)\right] = 0$$

$$0 \le r_1,r_2 \le N_a$$

$$(r_1,r_2) \neq (0,0) \tag{3.22}$$

We are interested only in stable solutions. These equations are trivially satisfied with $AB^*-A^*B=0$. The system of Equations (3.22) can collectively be rewritten

$$S(B,A)\ \mathrm{E}\left\{\begin{bmatrix} \frac{u(n_1,n_2)}{A^2} \\ \frac{u(n_1,n_2-1)}{A^2} \\ \vdots \\ \frac{u(n_1-1,n_2)}{A^2} \\ \frac{u(n_1-1,n_2-1)}{A^2} \\ \vdots \\ \frac{u(n_1-N_r,n_2-N_r)}{A^2} \end{bmatrix} \begin{bmatrix} \frac{u(n_1,n_2)}{AA^*} \cdots \frac{u(n_1-N_h,n_2-N_h)}{AA^*} \end{bmatrix} \begin{bmatrix} g(0,0) \\ \vdots \\ g(N_h,N_h) \end{bmatrix}\right\} = 0 \tag{3.23}$$

where

$$N_r = N_a + N_b \tag{3.24}$$

$$N_h = \max(N_a + N_b^*, N_a^* + N_b) \tag{3.25}$$

$$G(z_1,z_2) = AB^* - A^*B \tag{3.26}$$

$$S(B,A) = \begin{bmatrix} S_b \\ -- \\ S_a \end{bmatrix} \tag{3.27}$$

This can be modified to accommodate the general case in which the polynomials $A(q_1^{-1},q_2^{-1})$ and $B(q_1^{-1},q_2^{-1})$ are not relatively coprime (i.e., $A(q_1^{-1},q_2^{-1}) = \bar{A}(q_1^{-1},q_2^{-1})\mathrm{L}(q_1^{-1},q_2^{-1})$, $B(q_1^{-1},q_2^{-1}) = \bar{B}(q_1^{-1},q_2^{-1})\mathrm{L}(q_1^{-1},q_2^{-1})$ with $\bar{A}(q_1^{-1},q_2^{-1})$ and $\bar{B}(q_1^{-1},q_2^{-1})$ coprime). The matrix $S(B,A)$ is a block Sylvester matrix of dimension $\{(N_a+1)^2+(N_b+1)^2-1\}\mathrm{x}\{(N_a+N_b+1)^2\}$ and is defined as

$$S_b = \begin{bmatrix} 0 & b(0,0) & \cdots & 0 & 0 & b(1,0) & \cdots & 0 & \cdots & 0 & 0 & 0 & \cdots \\ 0 & 0 & b(0,0) & \cdots & 0 & 0 & b(1,0) & \cdots & \cdots & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & b(0,0) & b(0,1) & \cdots & 0 & b(1,0) & b(1,1) & \cdots & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & b(0,0) & b(0,1) & \cdots & 0 & b(1,0) & b(1,1) & \cdots & \end{bmatrix}$$

$$S_a = \begin{bmatrix} 1 & a(0,1) & \cdots & 0 & a(1,0) & a(1,1) & \cdots & 0 & \cdots & 0 & 0 & 0 & \cdots \\ 0 & 1 & a(0,1) & \cdots & 0 & a(1,0) & a(1,1) & \cdots & \cdots & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 1 & a(0,1) & \cdots & 0 & a(1,0) & a(1,1) & \cdots & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 1 & a(0,1) & \cdots & 0 & a(1,0) & a(1,1) & \cdots & \end{bmatrix} \tag{3.28}$$

Both Sylvester and block Sylvester matrices can be shown to have full rank if the corresponding numerator and denominator polynomials are coprime. Although both of these special cases must be addressed, the problem can be simplified if $S(B,A)$ can be factored out of the equation. In the 1-D case, the Sylvester matrix is square, but that is the case with 2-D filters only if N_b=0 (all pole filters). Otherwise, the matrix is underdetermined with a nontrivial null space. Let A_o and B_o represent a stationary point of the 2-D MSE surface. Again, if the adaptive filter is an AR model, that is, N_b=0, then $S(B_o,A_o)$ is square and nonsingular, and after factoring out the Sylvester matrix, (3.23) is equivalent to

$$0 = \left[\frac{1}{2\pi j}\right]^2 \oint_{c_1} \oint_{c_2} \frac{1}{A_o^2(z_1^{-1},z_2^{-1})} \frac{G(z_1,z_2)}{A_o(z_1,z_2)A^*(z_1,z_2)} \Phi_{uu}(z_1,z_1) z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2}$$

$$0 \le r_1, r_2 \le N_r \tag{3.29}$$

The line integrals above are evaluated on the distinguished boundary of the unit bidisc $T^2=\{(z_1,z_2):|z_1|=|z_2|=1\}$. The residue method can be used to evaluate the integral, but difficulties arise because the singularities of two-dimensional functions are algebraic functions. In the one-

dimensional case, Equation (3.22) can be written in the form $SVg=0$, or since the Sylvester matrix S is square and nonsingular, $Vg=0$. Now V is an MxN vanderMonde matrix constructed using the (distinct) poles of the one-dimensional analogue of (3.29) after evaluating the line integral. If the number of equations M is greater than or equal to the number of poles inside the unit circle of the 1-D version of (3.29), then the null space of V is empty. Therefore, $g=0$, which is equivalent to requiring that the integrand is analytic inside the unit circle. However, since the number of zeros inside the unit circle cannot match the number of poles inside the unit circle, the only remaining possibility is that $G(z)=0$, which is the uniqueness condition for 1-D IIR filters; i.e., $A^*(q^{-1})B(q^{-1})=B^*(q^{-1})A(q^{-1})$. Therefore, the condition for uniqueness requiring that the number of equations is greater than or equal to the number of poles inside the unit circle results in the familiar Soderstrom condition for uniqueness (if number of adaptive filter numerator coefficients is greater than or equal to the number of poles of the unknown filter, then a sufficient order system identification configuration has a unique minimum mean-squared error solution). The discussion above is abbreviated in that other conditions must be satisfied, such as the degenerated case and the case in which the denominator of the 1-D version of (3.29) has nondistinct poles. However, the result is the same for the general case.

There are four general categories under which these equations can be analyzed to determine unimodality conditions: 1) sufficient order adaptive filters with white noise, 2) sufficient order filters with colored noise, 3) insufficient order filters with white noise, and 4) insufficient order filters with colored noise. We next consider a specific example used previously in this chapter.

Example 1: Consider a system identification configuration with a zero-mean, unit-variance, white input signal. The fixed filter is separable with a first-order denominator and a constant numerator, and is given as

$$H^*(z_1,z_2) = \frac{D(z_1,z_2)}{C(z_1,z_2)} = \frac{d_o}{(1-p_1z_1^{-1})(1-p_2z_2^{-1})} \tag{3.30}$$

The fixed filter is to be identified by a general first-order adaptive filter of the form

$$H(z_1,z_2) = \frac{B(z_1,z_2)}{A(z_1,z_2)} = \frac{b_o}{1+az_1^{-1}+bz_2^{-1}+cz_1^{-1}z_2^{-1}} \tag{3.31}$$

with the denominator parameter set $\{a,b,c\}$ restricted to ensure stability [80] as

$$\frac{|1+a|}{|b+c|} > 1 \qquad \frac{|1-a|}{|b-c|} > 1 \qquad \text{or,}$$

$$\frac{|1+b|}{|a+c|} > 1 \qquad \frac{|1-b|}{|a-c|} > 1 \tag{3.32}$$

Experimental results from previous sections indicate that these conditions may result in a unimodal error surface, since no local minima were encountered with several different values of p_1 and p_2. This is, in fact, the case.

proof: For this case Equation (3.23) can be written

$$S(B,A)\ E\begin{bmatrix} \frac{u(n_1,n_2)}{A^2} \\ \frac{u(n_1-1,n_2)}{A^2} \\ \frac{u(n_1,n_2-1)}{A^2} \\ \frac{u(n_1-1,n_2-1)}{A^2} \end{bmatrix} \begin{bmatrix} \frac{u(n_1,n_2)}{AC} \cdots \frac{u(n_1-1,n_2-1)}{AC} \end{bmatrix} \begin{bmatrix} g(0,0) \\ g(1,0) \\ g(0,1) \\ g(1,1) \end{bmatrix} = \mathbf{0} \tag{3.33}$$

where

$$S(B,A) = \begin{bmatrix} 0 & b_o & 0 & 0 \\ 0 & 0 & b_o & 0 \\ 0 & 0 & 0 & b_o \\ 1 & a & b & c \end{bmatrix}$$

is nonsingular if $b_o \neq 0$. The case when $b_o = 0$ (degenerate) must be examined as a special case. To analyze the degenerate case, we must return to (3.22) and evaluate that equation. We begin

with the first case, $S(B, A)$ being nonsingular. Since $S(B, A)$ has only the trivial null space, (3.33) can be rewritten as (recalling that the input is white)

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{A(z_1,z_2)D(z_1,z_2)-C(z_1,z_2)B(z_1,z_2)}{A^2(z_1^{-1},z_2^{-1})A(z_1,z_2)C(z_1,z_2)} z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2} = 0$$

$$0 \le r_1, r_2 \le 1 \qquad (3.34)$$

since (3.33) is a cross-correlation.

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{D(z_1,z_2)}{A^2(z_1^{-1},z_2^{-1})C(z_1,z_2)} z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2} -$$

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{B(z_1,z_2)}{A^2(z_1^{-1},z_2^{-1})A(z_1,z_2)} z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2} = 0 \qquad (3.35)$$

First evaluating the line integrals in the left-hand term of (3.35) with $C(z_1,z_2)=(1-p_1 z_1^{-1})(1-p_2 z_2^{-1})$ and $|p_1|,|p_2| < 1$, we have

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{d_o z_1^{r_1} z_2^{r_2}}{A^2(z_1^{-1},z_2^{-1})(1-p_1 z_1^{-1})(1-p_2 z_2^{-1})} \frac{dz_1 dz_2}{z_1 z_2}$$

$$= \left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{d_o z_1^{r_1} z_2^{r_2}}{A^2(z_1^{-1},z_2^{-1})(z_1-p_1)(z_2-p_2)} dz_1 dz_2$$

$$= \left[\frac{1}{2\pi j}\right] \oint_{c_1} \frac{d_o z_1^{r_1}}{(z_1-p_1)} \left[\frac{1}{2\pi j}\right] \oint_{c_2} \frac{z_2^{r_2} dz_2}{A^2(z_1^{-1},z_2^{-1})(z_2-p_2)} dz_1$$

$$= \left[\frac{1}{2\pi j}\right] \oint_{c_1} \frac{d_o z_1^{r_1} p_2^{r_2}}{A^2(z_1^{-1},p_2^{-1})(z_1-p_1)} dz_1$$

$$= \frac{d_o p_1^{r_1} p_2^{r_2}}{A^2(p_1^{-1},p_2^{-1})} \qquad \text{for } 0 \le r_1, r_2 \le 1 \qquad (3.36)$$

$$\therefore \qquad \text{LHS} = \frac{d_o}{A^2(p_1^{-1},p_2^{-1})} \begin{bmatrix} 1 \\ p_1 \\ p_2 \\ p_1p_2 \end{bmatrix} \tag{3.37}$$

The evaluation of the right-hand term of Equation (3.35) is much more difficult, since the adaptive filter is not constrained to be separable. We must take this possibility into consideration and evaluate (3.35) for both a) the nonseparable and b) separable solution cases. First consider the evaluation of the right-hand side of (3.35) for case a) with a nonseparable solution.

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1} \oint_{c_2} \frac{b_o}{A^2(z_1^{-1},z_2^{-1})A(z_1,z_2)} z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2} \tag{3.38}$$

Even with simple 2-D polynomials, this line integral can be very tedious and difficult to evaluate. Algebraic manipulations allow us to first consider polynomials of z_1 with the coefficients themselves being functions of z_2, and then allow evaluation of two one-dimensional line integrals [80]. Consider

$$\begin{aligned} A(z_1,z_2) &= 1+ az_1^{-1} + bz_2^{-1} + cz_1^{-1}z_2^{-1} \\ z_1z_2A(z_1,z_2) &= z_1z_2 + az_2 + bz_1 + c \\ &= z_1[z_2+b] + [az_2+c] \\ &\text{let } f_0 = az_2+c \text{ and } f_1 = z_2+b \text{ so that} \\ z_1z_2A(z_1,z_2) &= f_1z_1 + f_0 \end{aligned} \tag{3.39}$$

and, similarly,

$$\begin{aligned} A(z_1^{-1},z_2^{-1}) &= 1+ az_1 + bz_2 + cz_1z_2 \\ &= g_0z_1 + g_1 \\ &\text{where } g_0 = +a+cz_2 \text{ and } g_1 = 1+bz_2 \end{aligned} \tag{3.40}$$

Since $A(z_1,z_2)$ is required to be stable, the singularity $z_1=-f_0/f_1$ lies within the z_1 unit circle for all $|z_2|=1$, and it is the only singularity enclosed by the contour. We can then evaluate the residue in (3.38) with respect to that single pole. First, (3.38) is written as

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{b_o z_1^{r_1} z_2^{r_2} dz_1 dz_2}{A^2(z_1^{-1},z_2^{-1})(f_1 z_1 + f_0)}$$

$$= \left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{b_o z_1^{r_1} z_2^{r_2} dz_1 dz_2}{A^2(z_1^{-1},z_2^{-1}) f_1 (z_1 + \frac{f_0}{f_1})}$$

$$= \left[\frac{1}{2\pi j}\right] \oint_{c_2} \frac{b_o \left[\frac{-f_0}{f_1}\right]^{r_1} z_2^{r_2}}{(g_0\frac{-f_0}{f_1}+g_1)^2 f_1} dz_2 = \left[\frac{1}{2\pi j}\right] \oint_{c_2} \frac{b_o \left[\frac{-f_0}{f_1}\right]^{r_1} z_2^{r_2} f_1}{(-g_0 f_0+g_1 f_1)^2} dz_2$$

(3.41)

At this point we require a short digression to examine the characteristics of the denominator polynomial $(-g_0f_0+g_1f_1)^2$. To proceed with the evaluation of the z_2 line integral, we must be able to identify zeros of the denominator which lie inside or outside of the z_2 unit circle. Fortunately, for first-order polynomials we can show that $-g_0f_0+g_1f_1$ has one zero inside the unit circle and one outside. The following facts are required:

i) $t_1 t_2 = 1$,

ii) $-g_0f_0+g_1f_1$ has only real roots, and

iii) $\beta > 0$

where t_1 and t_2 are the roots of the denominator polynomial $(-g_0f_0+g_1f_1)$, and $\beta=(b^2+1-a^2-c^2)$. Furthermore, define $\alpha=b-ac$. Expanding $(-g_0f_0+g_1f_1)$ we obtain

$$\begin{aligned} -g_0f_0+g_1f_1 &= (-a-cz_2)(az_2+c)+(1+bz_2)(z_2+b) \\ &= [-ac+b]z_2^2+[b^2+1-a^2-c^2]z_2+[-ac+b] \\ &= \alpha z_2^2 + \beta z_2 + \alpha \end{aligned}$$

$$= (z_2-t_1)(z_2-t_2) \tag{3.42}$$

where $$t_1, t_2 = \frac{-\beta \pm \sqrt{\beta^2-4\alpha^2}}{2\alpha} \tag{3.43}$$

Clearly, $t_1 t_2 = 1$, so that $t_1 = 1/t_2$ and i) is true. However, we must show that one pole is inside the unit circle and not on it. Facts ii) and iii) are required first.

$$\begin{aligned}\beta^2 - 4\alpha^2 &= (\beta-2\alpha)(\beta+2\alpha) = [b^2+1-a^2-c^2+2ac-2b][b^2+1-a^2-c^2-2ac+2b] \\ &= [(b-1)^2 - (a^2-2ac+c^2)][(b+1)^2 - (a+c)^2] \\ &= [(b-1)^2 - (a-c)^2][(b+1)^2 - (a+c)^2] > 0 \end{aligned} \tag{3.44}$$

with the last inequality guaranteed as a result of the stability conditions given in (3.32); therefore, both roots t_1 and t_2 are real (ii). To prove that $\beta > 0$, rewrite the stability conditions

$$\frac{|1+a|}{|b+c|} > 1 \qquad \frac{|1-a|}{|b-c|} > 1 \qquad \text{or,}$$

$$\frac{|1+b|}{|a+c|} > 1 \qquad \frac{|1-b|}{|a-c|} > 1 \tag{3.45}$$

From the first condition with $|a| < 1$

$$1-a^2 > |b^2-c^2|$$

if $b > c$

$$1-a^2 > b^2-c^2$$

$$\beta = [b^2+1-a^2-c^2] > 2(b^2-c^2) > 0$$

if $b < c$

$$1-a^2 > c^2-b^2$$

$$1-a^2-c^2+b^2=\beta>0$$

Therefore, $\beta>0$. Now, we must show

$$|t_1|=\left|\frac{-\beta+\sqrt{\beta^2-4\alpha^2}}{2\alpha}\right|<1$$

$$\left|-\beta+\sqrt{\beta^2-4\alpha^2}\right|<2|\alpha|$$

$$\left|\beta-\sqrt{\beta^2-4\alpha^2}\right|<2|\alpha|$$

$$2\sqrt{\beta^2-4\alpha^2}\left(\sqrt{\beta^2-4\alpha^2}-\beta\right)<0 \qquad (3.46)$$

which gives

$$\sqrt{\beta^2-4\alpha^2}<\beta \qquad (3.47)$$

for all $\beta>0$. With this observation, the result has been proven that $|t_1|<1$; therefore, one pole lies within the unit circle and one lies outside the unit circle.

Continuing the evaluation of (3.41)

$$\text{RHS}=\left[\frac{1}{2\pi j}\right]\oint_{c_2}\frac{b_o\left[\frac{-f_0}{f_1}\right]^{r_1}z_2^{r_2}f_1}{(\alpha z_2^2+\beta z_2+\alpha)^2}dz_2=\left[\frac{1}{2\pi j}\right]\oint_{c_2}\frac{b_o f_0^{r_1}z_2^{r_2}f_1^{1-r_1}}{(\alpha z_2^2+\beta z_2+\alpha)^2}dz_2$$

$$=(z_2^{r_2}f_0^{r_2}f_1^{1-r_2})D\left[\frac{b_o}{(z_2-t_2)^2}\right]+D(z_2^{r_2}f_0^{r_2}f_1^{1-r_2})\left[\frac{b_o}{(z_2-t_2)^2}\right]\quad\text{with } z_2=t_1 \qquad (3.48)$$

Combining these results into a matrix equation, we have

$$\begin{bmatrix} f_1 & 1 & 1 \\ z_2 f_1 & 2z_2+b & p_2 \\ f_0 & a & p_1 \\ z_2 f_0 & 2az_2+c & p_1 p_2 \end{bmatrix} \begin{bmatrix} \mathrm{D}\left[\frac{b_o}{(z_2-t_2)^2}\right] \\ \left[\frac{b_o}{(z_2-t_2)^2}\right] \\ \frac{d_o}{A^2(p_1^{-1},p_2^{-1})} \end{bmatrix} = \mathbf{0} \tag{3.49}$$

$$\boldsymbol{V} = \begin{bmatrix} f_1 & 1 & 1 \\ z_2 f_1 & 2z_2+b & p_2 \\ f_0 & a & p_1 \\ z_2 f_0 & 2az_2+c & p_1 p_2 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & p_2-z_2 \\ f_0 & a & p_1 \\ 0 & f_0 & p_1 p_2 - z_2 p_1 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & a & p_1 \\ 0 & f_0 & dp_1 \end{bmatrix} \tag{3.50}$$

where $d=p_2-z_2$, noting also that for a nonseparable denominator polynomial ($ab \neq c$) we have one of the following three possibilities: i) $f_0=0, f_1 \neq 0$, ii) $f_0 \neq 0, f_1=0$, and iii) $f_0 \neq 0, f_1 \neq 0$. We must evaluate the rank of this matrix for each of these three cases, and also consider $d=0$ or $d \neq 0$. Also assume $p_1, p_2 \neq 0$.

i) $f_0=0, f_1 \neq 0$. For $a=0$ and $d=0$

$$\boldsymbol{V} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & 0 \\ 0 & 0 & p_1 \\ 0 & 0 & 0 \end{bmatrix}$$

Clearly rank $\boldsymbol{V} = 3$. If $d \neq 0$,

$$\boldsymbol{V} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ 0 & 0 & p_1 \\ 0 & 0 & dp_1 \end{bmatrix}$$

rank $\boldsymbol{V} = 3$ also. If $a \neq 0$ and $d=0$

$$\boldsymbol{V} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & 0 \\ 0 & a & p_1 \\ 0 & 0 & 0 \end{bmatrix}$$

rank $V = 3$ also. If $a \neq 0$ and $d \neq 0$,

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ 0 & a & p_1 \\ 0 & 0 & dp_1 \end{bmatrix}$$

rank $V = 3$ also.

ii) $f_0 \neq 0, f_1 = 0$. For $a = 0$ and $d = 0$,

$$V \rightarrow \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ f_0 & 0 & p_1 \\ 0 & f_0 & 0 \end{bmatrix}$$

rank $V = 3$. If $a = 0$ and $d \neq 0$,

$$V \rightarrow \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & d \\ f_0 & 0 & p_1 \\ 0 & f_0 & dp_1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & d \\ f_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

rank $V = 3$. If $a \neq 0$ and $d = 0$,

$$V \rightarrow \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ f_0 & a & p_1 \\ 0 & f_0 & 0 \end{bmatrix}$$

rank $V = 3$. If $a \neq 0$ and $d \neq 0$,

$$V \rightarrow \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & d \\ f_0 & a & p_1 \\ 0 & f_0 & dp_1 \end{bmatrix}$$

rank $V = 3$.

iii) $f_0 \neq 0, f_1 \neq 0$. For $a=0$ and $d=0$,

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & 0 \\ f_0 & 0 & p_1 \\ 0 & f_0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 0 & 1 \\ 0 & f_1 & 0 \\ f_0 & 0 & p_1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 0 & 1 \\ 0 & f_1 & 0 \\ 0 & 0 & p_1\text{-}f_0/f_1 \\ 0 & 0 & 0 \end{bmatrix}$$

now the matrix is either rank 3, or rank 2, with no solution in the dimension-one null space. For $a=0$ and $d \neq 0$,

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & 0 & p_1 \\ 0 & f_0 & dp_1 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & 0 & p_1 \\ 0 & 0 & d(p_1\text{-}f_0/f_1) \end{bmatrix}$$

If $p_1\text{-}f_0/f_1 \neq 0$, rank = 3. With $p_1\text{-}f_0/f_1 = 0$

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & 0 & p_1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 1\text{-}p_1f_1/f_o \\ 0 & f_1 & d \\ f_0 & 0 & p_1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 0 \\ 0 & f_1 & d \\ f_0 & 0 & p_1 \\ 0 & 0 & 0 \end{bmatrix}$$

which has rank 3. For $a \neq 0$ and $d=0$,

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & 0 \\ f_0 & a & p_1 \\ 0 & f_0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 0 & 1 \\ 0 & f_1 & 0 \\ f_0 & 0 & p_1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 0 & 1 \\ 0 & f_1 & 0 \\ 0 & 0 & p_1\text{-}f_0/f_1 \\ 0 & 0 & 0 \end{bmatrix}$$

which has no solution. And finally, with $a \neq 0$ and $d \neq 0$,

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & a & p_1 \\ 0 & f_0 & dp_1 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & a & p_1 \\ 0 & 0 & d(p_1\text{-}f_0/f_1) \end{bmatrix}$$

As above, if $p_1\text{-}f_0/f_1 \neq 0$ rank = 3. If $p_1\text{-}f_0/f_1 = 0$, then

$$V \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ f_0 & a & p_1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ 0 & a\text{-}f_0/f_1 & p_1\text{-}f_0/f_1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} f_1 & 1 & 1 \\ 0 & f_1 & d \\ 0 & a\text{-}f_0/f_1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

which has rank 3 since $a \neq f_0/f_1$ for a nonseparable polynomial. We have now shown that no nonseparable solutions exist.

Now we evaluate (3.35) for case b), the separable solution. The right-hand term of (3.35) is repeated

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1} \oint_{c_2} \frac{B(z_1,z_2)}{A^2(z_1^{-1},z_2^{-1})A(z_1,z_2)} z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2} \tag{3.51}$$

With $A(z_1,z_2) = (1\text{-}x_1 z_1^{-1})(1\text{-}x_2 z_2^{-1})$ this becomes

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1} \oint_{c_2} \frac{B(z_1,z_2)}{A^2(z_1^{-1},z_2^{-1})(1\text{-}x_1 z_1^{-1})(1\text{-}x_2 z_2^{-1})} z_1^{r_1} z_2^{r_2} \frac{dz_1 dz_2}{z_1 z_2} \tag{3.52}$$

from which it can easily be shown that (3.35) is equivalent to

$$\begin{bmatrix} 1 & 1 \\ x_1 & p_1 \\ x_2 & p_2 \\ x_1x_2 & p_1p_2 \end{bmatrix} \begin{bmatrix} \dfrac{-b_o}{A^2(x_1^{-1},x_2^{-1})} \\ \dfrac{d_o}{A^2(p_1^{-1},p_2^{-1})} \end{bmatrix} = \mathbf{0} \tag{3.53}$$

The only solution to this equation is $x_1=p_1$ and $x_2=p_2$, with $b_o=d_o$. We have to only examine the degenerate case to conclude uniqueness. With $B(q_1^{-1},q_2^{-1})=0$, Equation (3.22) becomes

$$E\left[\frac{D(q_1^{-1},q_2^{-1})}{C(q_1^{-1},q_2^{-1})}u(n_1,n_2)\ \frac{1}{A(q_1^{-1},q_2^{-1})}u(n_1-r_1,n_2-r_2)\right]=0$$

$$0 \le r_1,r_2 \le N_b \tag{3.54}$$

Since $N_b=0$, we have one equation. This can be rewritten as

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{D(z_1,z_2)}{A(z_1^{-1},z_2^{-1})C(z_1,z_2)}\ \frac{dz_1dz_2}{z_1z_2} = 0 \tag{3.55}$$

$$\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1}\oint_{c_2} \frac{d_o}{A(z_1^{-1},z_2^{-1})(z_1-p_1)(z_2-p_2)}\ dz_1dz_2 = 0 \tag{3.56}$$

Evaluating (3.56) gives

$$\frac{d_o}{A(p_1^{-1},p_2^{-1})}=0 \tag{3.57}$$

which is impossible since we require $d_o\neq 0$. Therefore, no degenerate solution exists, which proves unimodality for the example under consideration. Even for this relatively simple case the solution is very complex.

Several unimodal conditions are readily available for one-dimensional IIR adaptive filters. However, because of the difficulties associated with evaluating two-dimensional line integrals, applying similar solution methods to the problem at hand is futile. It is possible to analyze some very simple special cases as we have shown. When the MSE surface has degenerate solutions, it is possible to present some general results without evaluating the line integrals introduced earlier.

The following theorem gives a sufficient but not necessary condition for the existence of multiple stationary points.

Theorem: For a 2-D IIR adaptive filter, the existence of a stable degenerate ($B(q_1^{-1},q_2^{-1})=0$) stationary point on the MSE surface implies multimodality if $N_a > N_b = 0$.

Proof: We first show that stationary points are saddle points. This result is valid regardless of the size of the numerator. The proof follows the work on 1-D filters in [42] and [43]. The 2-D MSE with an arbitrary numerator polynomial $b(n_1,n_2)$ is

$$W(\boldsymbol{a},\boldsymbol{b}) = \frac{1}{2}\mathrm{E}[e^2(n_1,n_2)] = \frac{1}{2}\mathrm{E}\left[\left\{\left(\frac{B^*}{A^*} - \frac{B}{A}\right)(q_1^{-1},q_2^{-1})\, u(n_1,n_2)\right\}^2\right] + \mathrm{E}[v^2(n_1,n_2)] \tag{3.58}$$

$$W(\boldsymbol{a},\boldsymbol{b}) = \frac{1}{2}F_0 - \boldsymbol{F}_1(\boldsymbol{a})^{\mathrm{T}}\boldsymbol{b} + \frac{1}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}_2(\boldsymbol{a})\boldsymbol{b} \tag{3.59}$$

where

$$F_0 = \mathrm{E}\left[\left\{\frac{B^*(q_1^{-1},q_2^{-1})}{A^*(q_1^{-1},q_2^{-1})}u(n_1,n_2)\right\}^2\right] + \mathrm{E}[v^2(n_1,n_2)] \tag{3.60}$$

allowing for an additive noise source $v(n_1,n_2)$. Also, the length $(N_b+1)^2$ vector $\boldsymbol{F}_1(\boldsymbol{a})$ and the $(N_b+1)^2$x$(N_b+1)^2$ autocorrelation matrix $\boldsymbol{F}_2(\boldsymbol{a})$ are defined as follows

$$F_1(\boldsymbol{a})_{iN_b+j} = \mathrm{E}\left[\frac{B^*(q_1^{-1},q_2^{-1})}{A^*(q_1^{-1},q_2^{-1})}u(n_1,n_2)\right]\left[\frac{q_1^{-i}q_2^{-j}}{A(q_1^{-1},q_2^{-1})}u(n_1,n_2)\right] \tag{3.61}$$

$$F_2(\boldsymbol{a})_{iN_b+j,kN_b+l} = \mathrm{E}\left[\frac{q_1^{-i}q_2^{-j}}{A(q_1^{-1},q_2^{-1})}u(n_1,n_2)\right]\left[\frac{q_1^{-k}q_2^{-l}}{A(q_1^{-1},q_2^{-1})}u(n_1,n_2)\right] \tag{3.62}$$

with

$$\boldsymbol{b} = [b(0,0)\ b(0,1)\ \ldots\ b(0,N_b)\ldots\ b(N_b,N_b)]^T$$

By simply examining the behavior of this function around and near a stable degenerate stationary point, the theorem can be proven. Let $(\boldsymbol{a}_o,0)$ be such a point. Then

$$W(\boldsymbol{a}_o,0) = \frac{1}{2}F_0 \tag{3.63}$$

and a point $(\boldsymbol{a}_o,\delta\boldsymbol{b})$ arbitrarily close to the stationary point gives a larger value of the loss function for any $\delta\boldsymbol{b}$.

$$W(\boldsymbol{a}_o,\delta\boldsymbol{b}) = \frac{1}{2}F_0 + \frac{1}{2}\,\delta\boldsymbol{b}^T\boldsymbol{F}_2(\boldsymbol{a}_o)\delta\boldsymbol{b} > \frac{1}{2}F_0 = W(\boldsymbol{a}_o,0) \tag{3.64}$$

since $\boldsymbol{F}_1(\boldsymbol{a}_o) = 0$ for degenerate solutions ($\boldsymbol{b}(\boldsymbol{a}_o)=0$ implies $\boldsymbol{F}_1(\boldsymbol{a}_o)=0$), and $\boldsymbol{F}_2(\boldsymbol{a}_o)$ is positive definite. Recognizing that if we consider the point $(\boldsymbol{a}_o+\delta\boldsymbol{a},\boldsymbol{b}(\boldsymbol{a}_o+\delta\boldsymbol{a}))$ with $\boldsymbol{b}(\boldsymbol{a}_o+\delta\boldsymbol{a})=\boldsymbol{F}_2(\boldsymbol{a}_o+\delta\boldsymbol{a})^{-1}\boldsymbol{F}_1(\boldsymbol{a}_o+\delta\boldsymbol{a})$ satisfying the minimization of $W(\boldsymbol{a},\boldsymbol{b})$ with respect to $\boldsymbol{b}$, then

$$W(\boldsymbol{a}_o+\delta\boldsymbol{a},\boldsymbol{b}(\boldsymbol{a}_o+\delta\boldsymbol{a})) = \frac{1}{2}F_0 - \boldsymbol{F}_1(\boldsymbol{a}_o+\delta\boldsymbol{a})^T\boldsymbol{b} + \frac{1}{2}\boldsymbol{b}^T\boldsymbol{F}_2(\boldsymbol{a}_o+\delta\boldsymbol{a})\boldsymbol{b}$$

$$= \frac{1}{2}F_0 - \boldsymbol{F}_1(\boldsymbol{a}_o+\delta\boldsymbol{a})^T\boldsymbol{F}_2(\boldsymbol{a}_o+\delta\boldsymbol{a})^{-1}\boldsymbol{F}_1(\boldsymbol{a}_o+\delta\boldsymbol{a}) + \frac{1}{2}\boldsymbol{b}^T\boldsymbol{F}_2(\boldsymbol{a}_o+\delta\boldsymbol{a})\boldsymbol{b}$$

$$= \frac{1}{2}F_0 - \frac{1}{2}\boldsymbol{F}_1(\boldsymbol{a}_o+\delta\boldsymbol{a})^T\boldsymbol{F}_2(\boldsymbol{a}_o+\delta\boldsymbol{a})^{-1}\boldsymbol{F}_1(\boldsymbol{a}_o+\delta\boldsymbol{a}) < \frac{1}{2}F_0 \tag{3.65}$$

Therefore, since the loss function $W(\boldsymbol{a}_o,\delta\boldsymbol{b})$ increases from the stationary point for $\delta\boldsymbol{b}$ not satisfying $\delta\boldsymbol{b}=\boldsymbol{F}_2(\boldsymbol{a}_o)^{-1}\boldsymbol{F}_1(\boldsymbol{a}_o)$, i.e., not satisfying the minimization with respect to $\boldsymbol{b}$, while simultaneously decreasing for any $W(\boldsymbol{a}_o+\delta\boldsymbol{a},\boldsymbol{b}(\boldsymbol{a}_o+\delta\boldsymbol{a}))$ along the path of $\boldsymbol{b}(\boldsymbol{a}_o+\delta\boldsymbol{a})$ as shown above, we conclude that stable degenerate stationary points of the 2D IIR MSE surface are saddle

points. Furthermore, if the numerator polynomial is of order 0, i.e., $B(q_1^{-1},q_2^{-1})=b(0,0)$, then the reduced coefficient space is partitioned at the degenerate point $(a_o,b(0,0)=0)$ with each partition $(a_o+\delta a,b(a_o+\delta a))$ and $(a_o-\delta a,b(a_o+\delta a))$ containing a local minimum. This observation completes the proof of the theorem.

Experimental evidence supports the following modified 2-D extension of Stearns' conjecture:

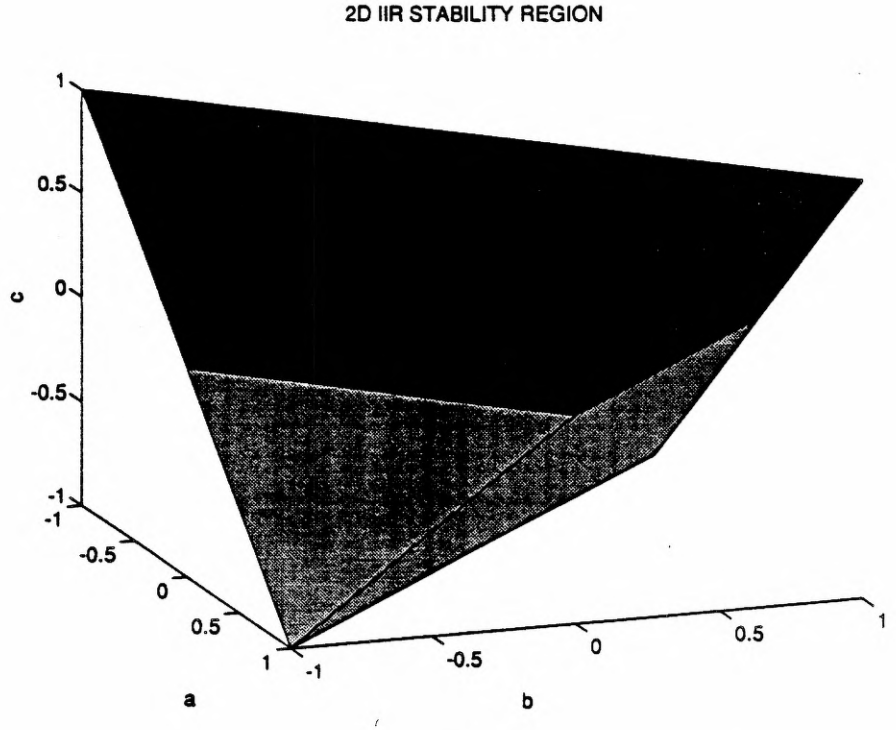
Conjecture: First- and second- (sufficient) order two-dimensional IIR adaptive filters with white noise have a unimodal MSE surface, with the possible exception of the over-parameterized case with $N_b^*=0$ and $N_a^*=2$.

For the case of a 2-D IIR adaptive filter with a first-order denominator, we can graphically examine error surface characteristics [81],[82]. The general expression for the mean squared error of a 2-D adaptive filter in system identification is

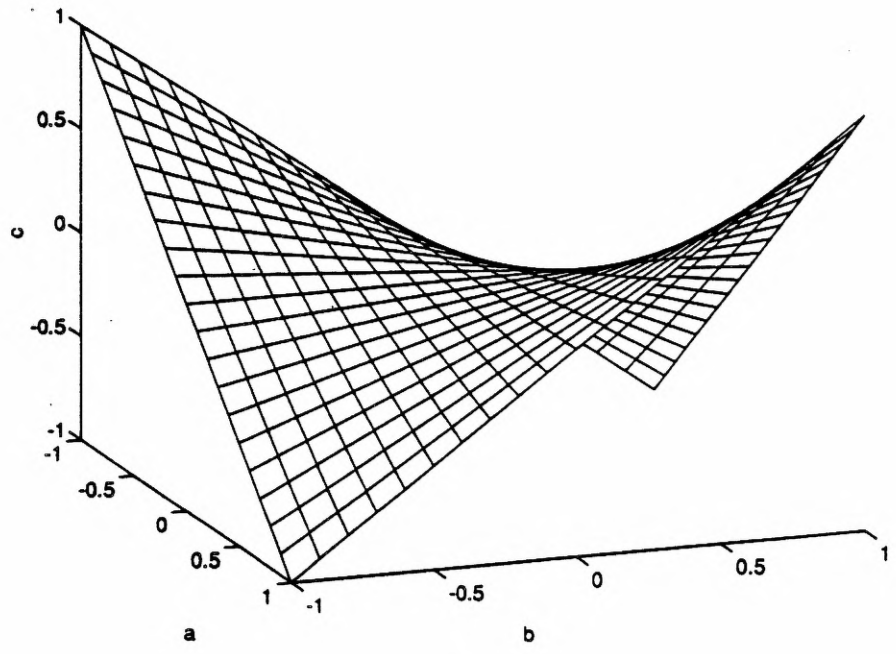
$$\xi = \phi_{dd}(0,0) + \left[\frac{1}{2\pi j}\right]^2 \oint_{c_1} \oint_{c_2} H(z_1^{-1},z_2^{-1})\Phi_{uu}(z_1,z_1)H(z_1,z_2)\,\frac{dz_1 dz_2}{z_1 z_2}$$

$$-\left[\frac{1}{2\pi j}\right]^2 \oint_{c_1} \oint_{c_2} 2H^*(z_1^{-1},z_2^{-1})\Phi_{uu}(z_1,z_1)H(z_1,z_2)\,\frac{dz_1 dz_2}{z_1 z_2} \qquad (3.66)$$

Example 2: Consider the unimodal configuration of example 1. Since the adaptive filter has only three denominator coefficients, we can plot constant-MSE contours in the three-dimensional parameter space. Regardless of the order of the numerator, we can reduce the dimensionality of the problem to three by first optimizing with respect to the numerator coefficients. In Figures 3.22a) and 3.22b) we show the stability region for the first-order denominator polynomial and the separable coefficient subspace. In Figures 3.23 and 3.24 we show the reduced, normalized error contours for example 1 ($p_1=p_2=0.1$) with the MSE=0.1 and MSE=0.9, respectively. The contours are converging around the unique stationary point (a,b,c) = (-0.1, -0.1, 0.01).



a)



b)

Figure 3.22 a) Stability region of the first-order 2-D IIR filter.
b) Separable stability region of the first-order 2-D IIR filter.

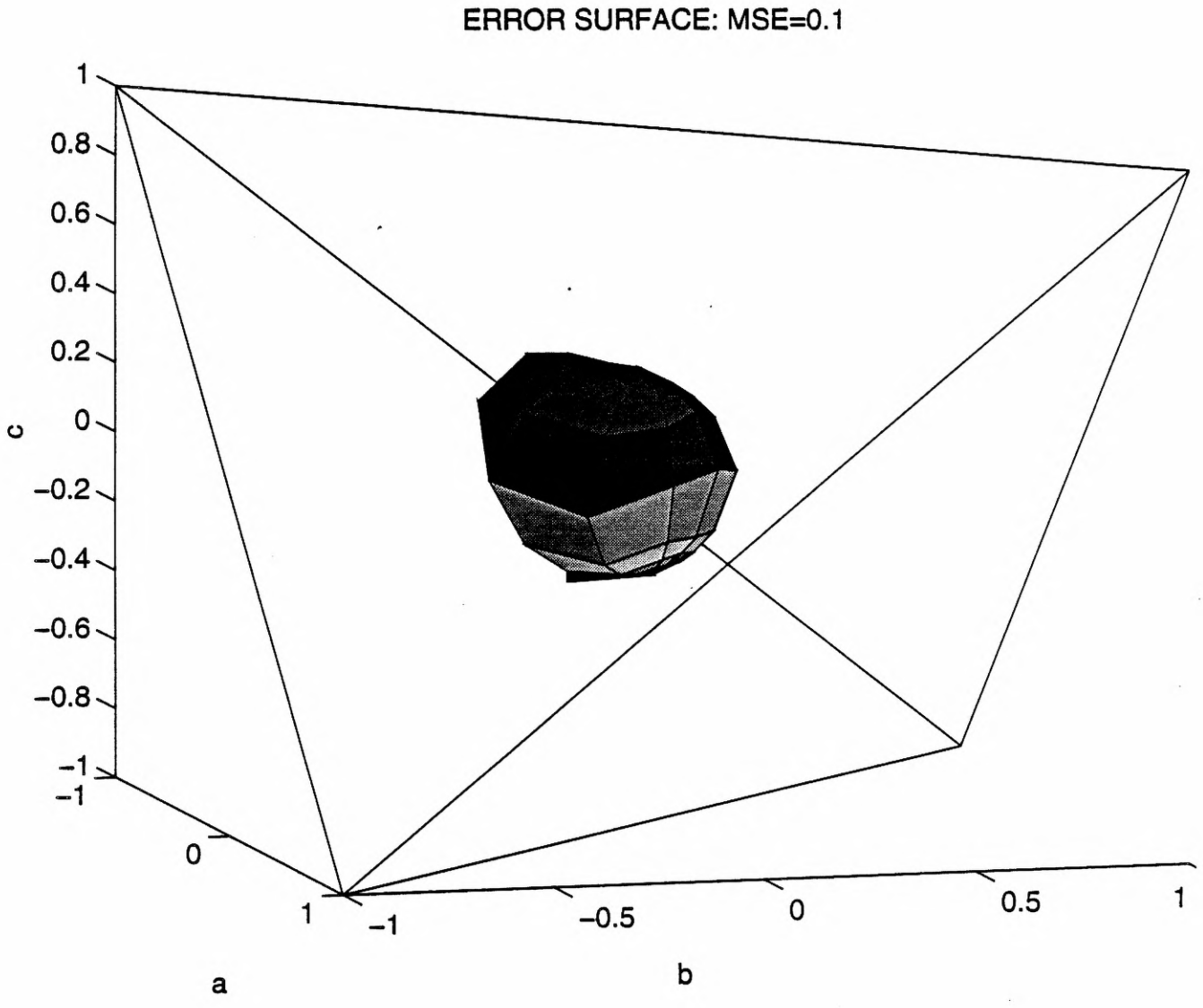


Figure 3.23 Reduced, normalized error contour for the 2-D IIR adaptive filter from example 1 with MSE=0.1 (example 2).

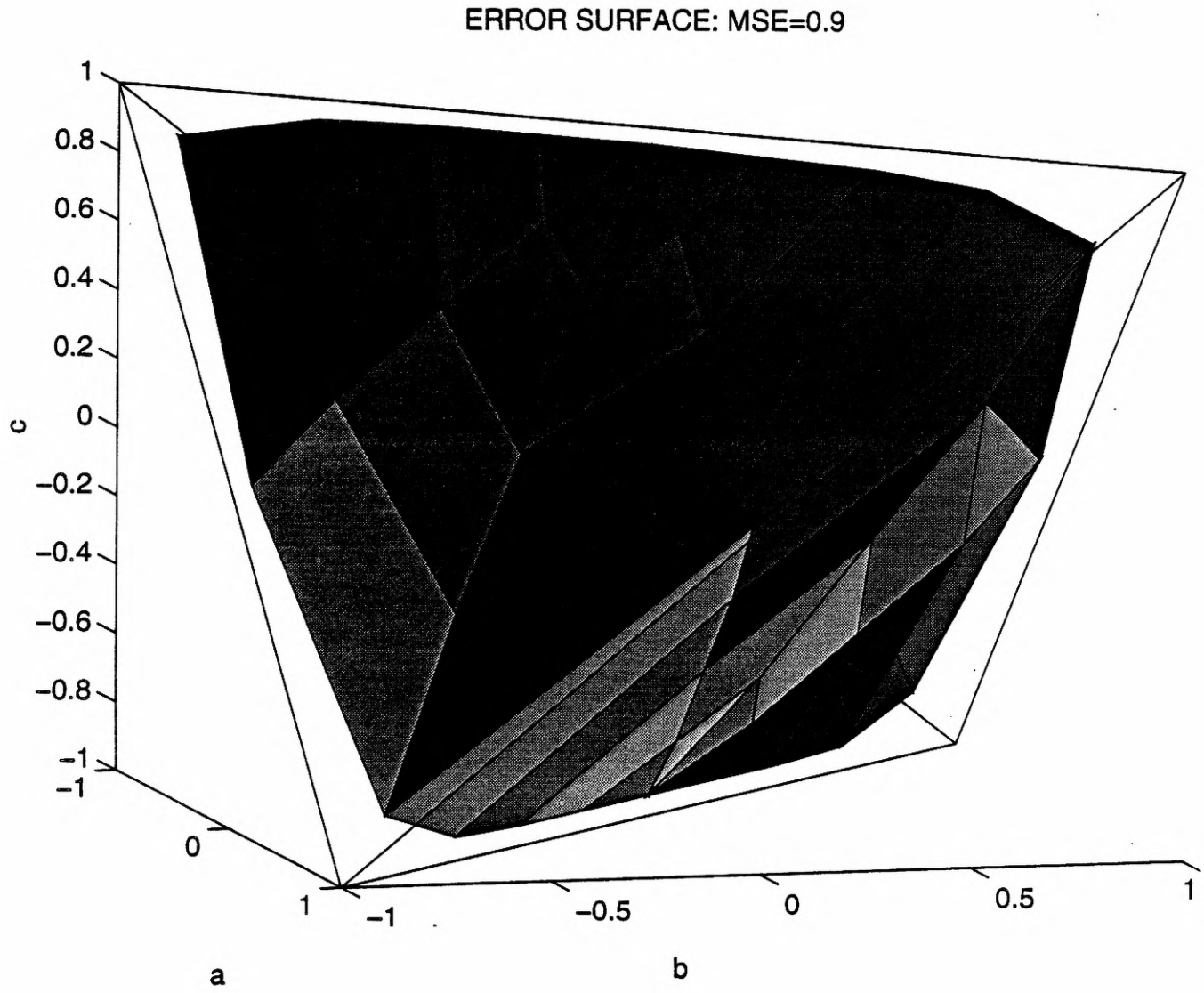


Figure 3.24 Reduced, normalized error contour for the 2-D IIR adaptive filter in example 1 with MSE=0.9 (example 2).

Example 3: Next we consider the same adaptive filter as in example 1, but we construct a separable model using the 1-D prototype from Nayeri et al.[43], example 1, so that the adaptive filter is of insufficient order to identify the model filter. In this case the 1-D prototype has a first-order numerator and a second-order denominator, which was shown by Nayeri et al. to produce a multimodal 1-D error surface (reproduced in Figure 3.25). Our 2-D configuration also produced a multimodal error surface for this example, which is shown in Figures 3.26 and 3.27 for MSE=0.98 and MSE=.95, respectively. The existence of separate, distinct surfaces in the three-dimensional coefficient space at any particular contour level proves that a local minimum is present. This was expected, since it also can be shown that the 2-D structural configuration for this example does have a degenerate stationary solution, which by the theorem guarantees multiple minima. The next example demonstrates that the theorem is a sufficient but not necessary condition for the existence of multiple minima.

Example 4: Consider again the adaptive filter from example 1. Now it is used again as an insufficient order model to identify the following fixed (separable) filter with a white input signal

$$H^*(z_1,z_2) = \frac{(1-0.85z_1^{-1})(1-0.85z_2^{-1})}{(1-0.99z_1^{-1})(1-0.99z_2^{-1})}$$

The error surface for the corresponding 1-D prototype is shown in Figure 3.28. No stable, degenerate solution exists for either the 1-D or 2-D case. However, the error contour plotted in Figure 3.29 for MSE=0.77 shows a local minima. Therefore the condition given in the theorem is not a necessary condition.

Example 5: Again using the same adaptive filter structure from above, we configure the following nonseparable fixed filter in system identification

$$H^*(z_1,z_2) = \frac{2}{1-0.4z_1^{-1}-0.4z_2^{-1}}$$

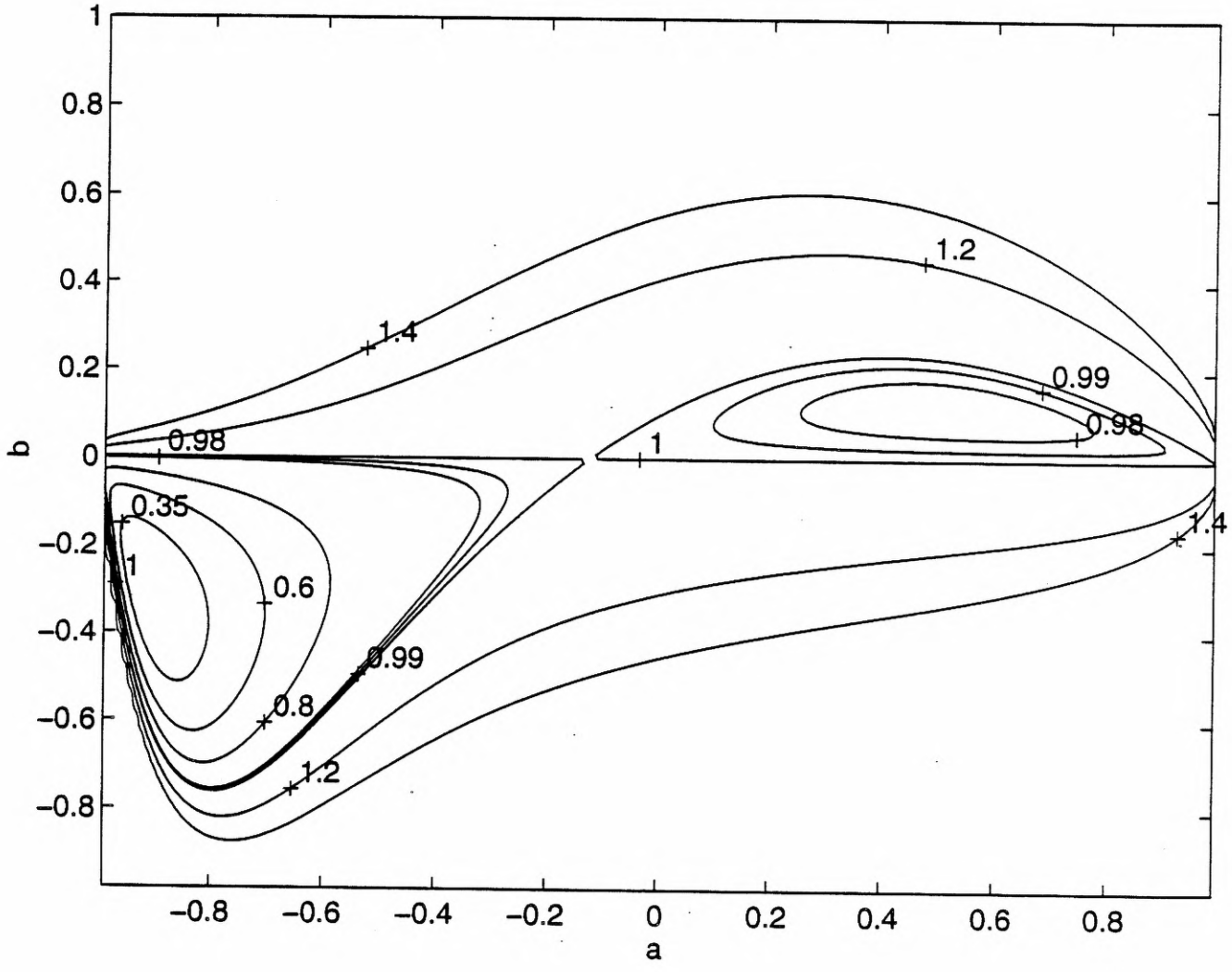


Figure 3.25 Normalized error contours for the 1-D IIR prototype adaptive filter (example 3).

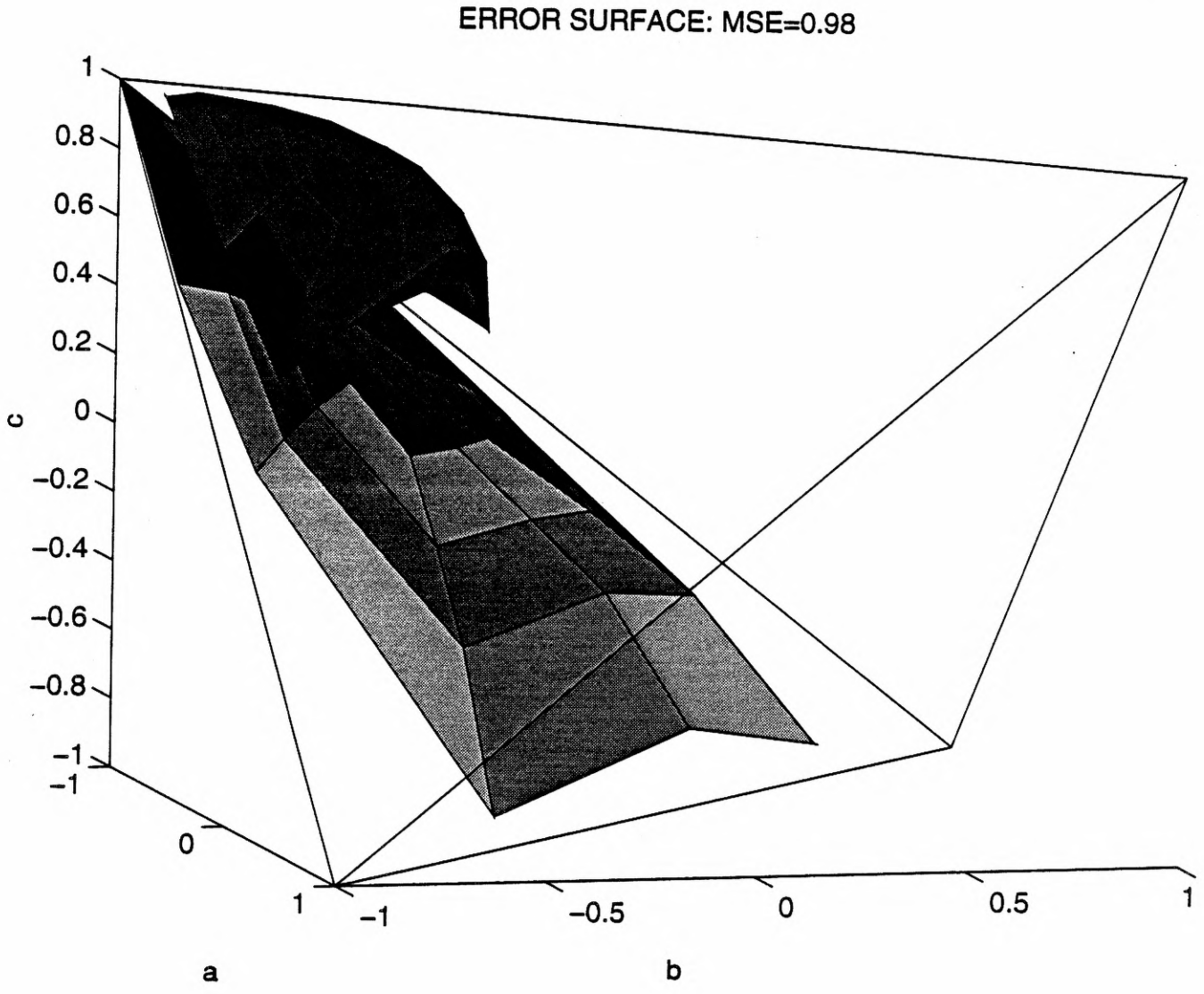


Figure 3.26 Reduced, normalized error contour for the insufficient, first-order 2-D IIR adaptive filter with MSE=0.98 (example 3).

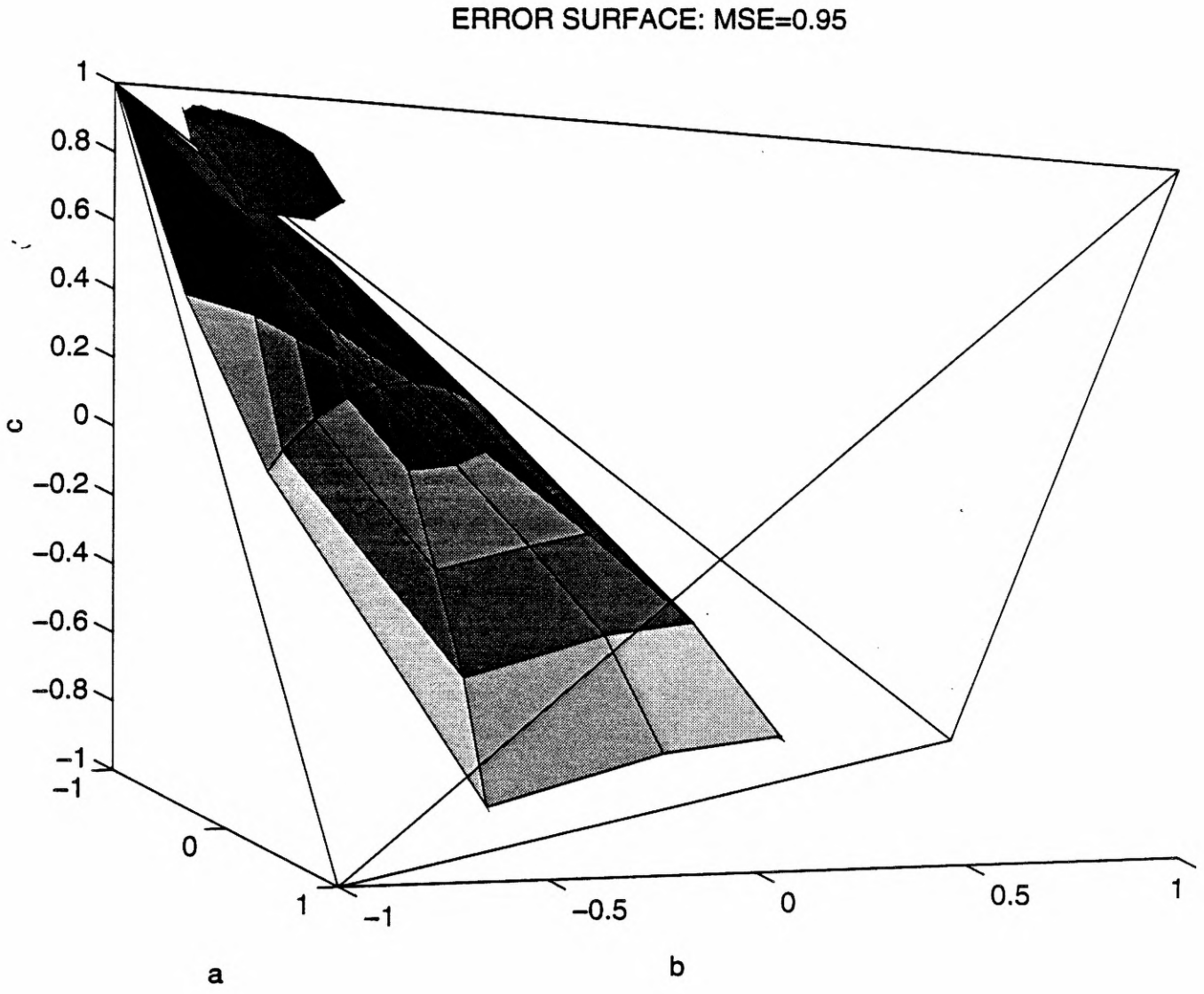


Figure 3.27 Reduced, normalized error contour for the insufficient, first-order 2-D IIR adaptive filter with MSE=0.95 (example 3).

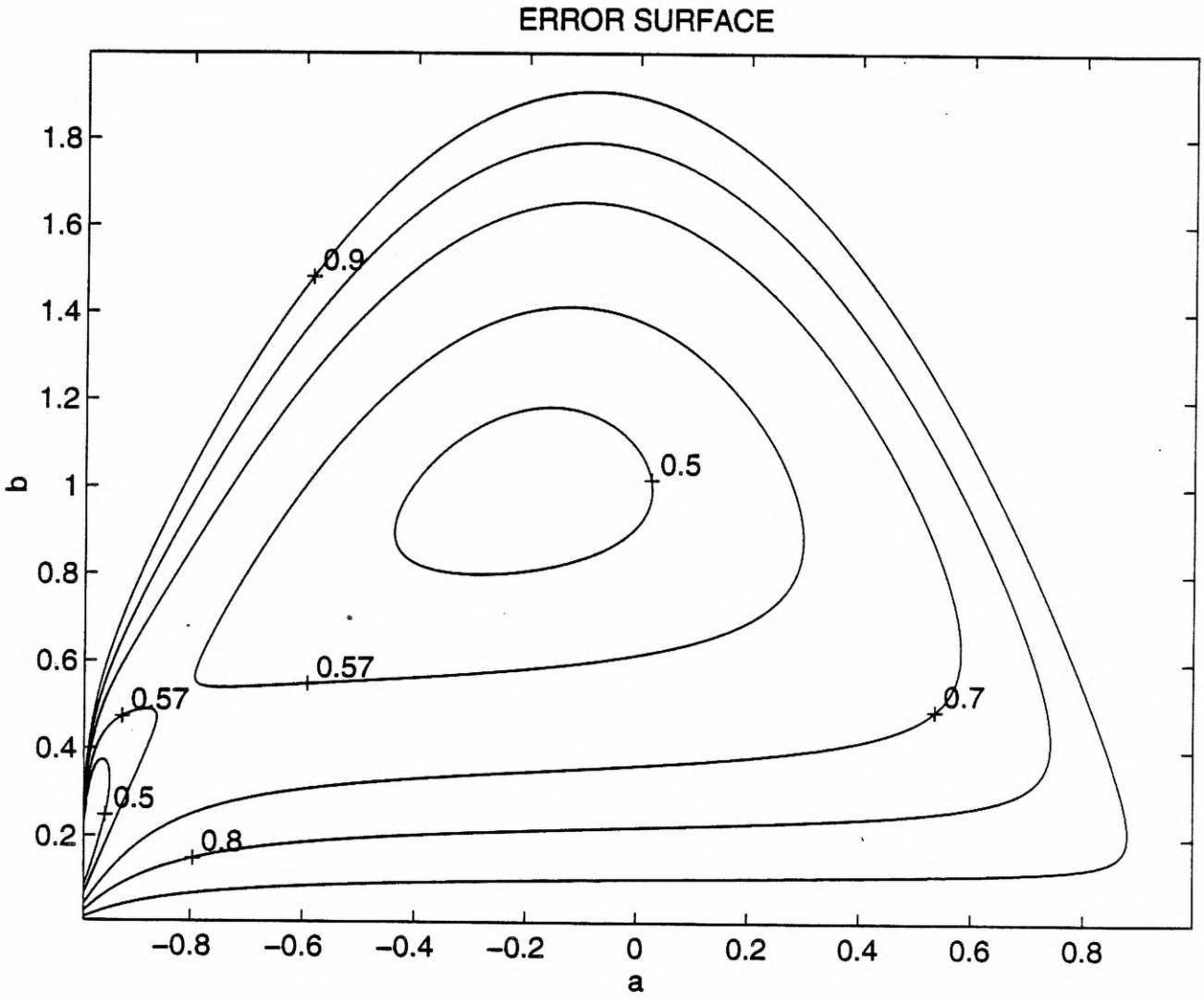


Figure 3.28 Normalized error contours for the 1-D IIR prototype adaptive filter (example 4).

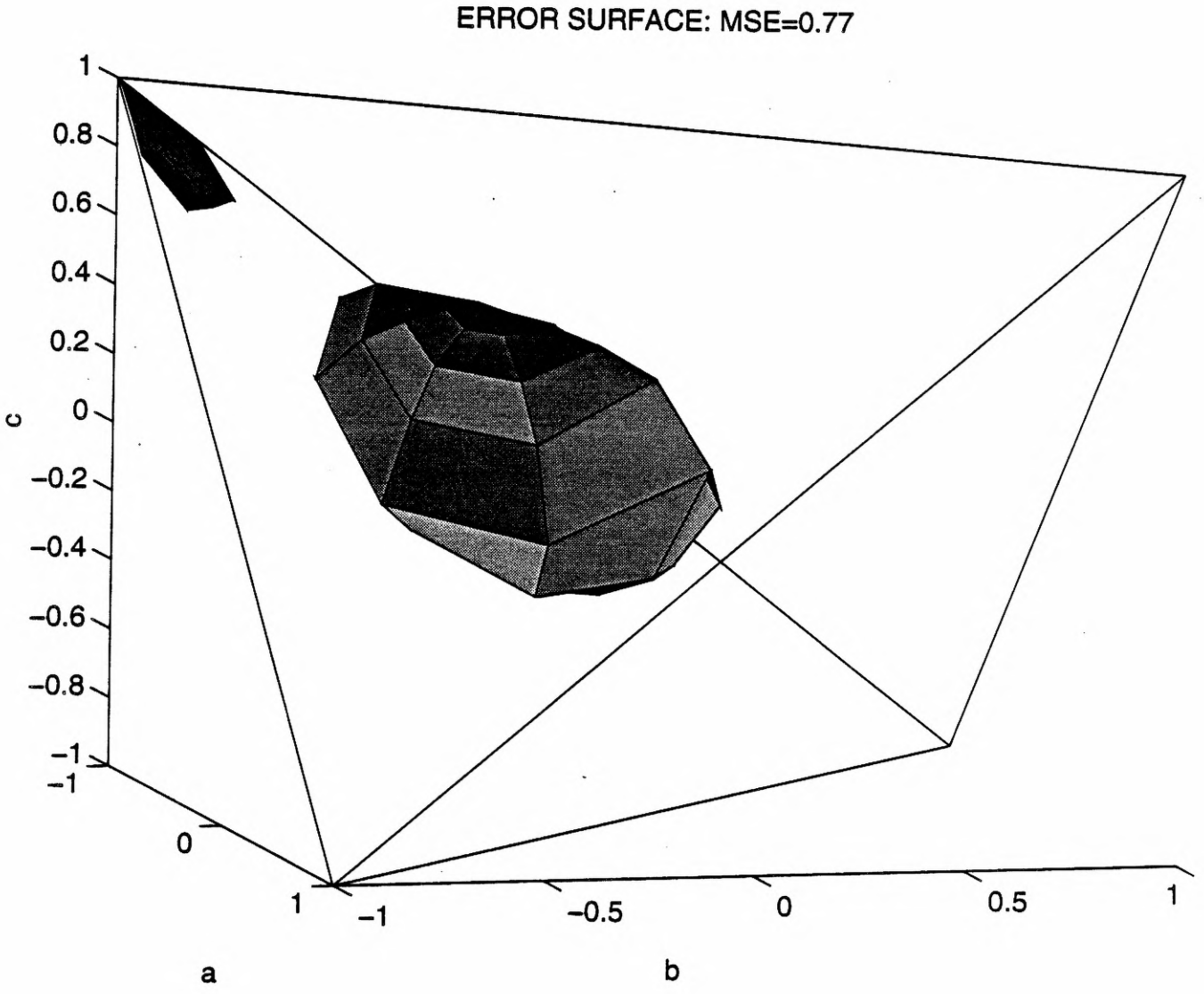


Figure 3.29 Reduced, normalized error contour for the insufficient, first-order 2-D IIR adaptive filter with MSE=0.77 (example 4).

Neither the adaptive filter nor the fixed filter are separable, and the adaptive filter is of sufficient order to model the fixed structure. Figure 3.30 shows error surface contours for MSE=0.95, 0.9, 0.5 and 0.1. The plots do not show any local minima since no separate distinct surfaces are evident. It can be shown that this structural configuration has no degenerate, stable solution.

Example 6: We present here one additional example using the same first-order adaptive filter as above. The fixed filter is nonseparable.

$$H^*(z_1,z_2) = \frac{1}{1+0.2z_1^{-1}+0.2z_2^{-1}+0.5z_1^{-1}z_2^{-1}}$$

The adaptive filter is again of sufficient order to model the fixed filter, and the error contours are plotted in Figure 3.31 for MSE=0.98, 0.9, 0.5 and 0.1. As in the previous example, the error surface here is unimodal, and this case clearly has no degenerate solutions.

Example 7: Now consider the following nonseparable adaptive filter

$$H(z_1,z_2) = \frac{1+dz_1^{-1}+ez_2^{-1}+fz_1^{-1}z_2^{-1}}{1+az_1^{-1}+bz_2^{-1}+cz_1^{-1}z_2^{-1}}$$

which identifies the matched order fixed filter

$$H^*(z_1,z_2) = \frac{1+0.5z_1^{-1}+0.5z_2^{-1}+0.1z_1^{-1}z_2^{-1}}{1+0.2z_1^{-1}+0.2z_2^{-1}+0.5z_1^{-1}z_2^{-1}}$$

The error contours are plotted in Figure 3.32 for MSE=0.1 and 0.2. No local minima are evident in this case, either.

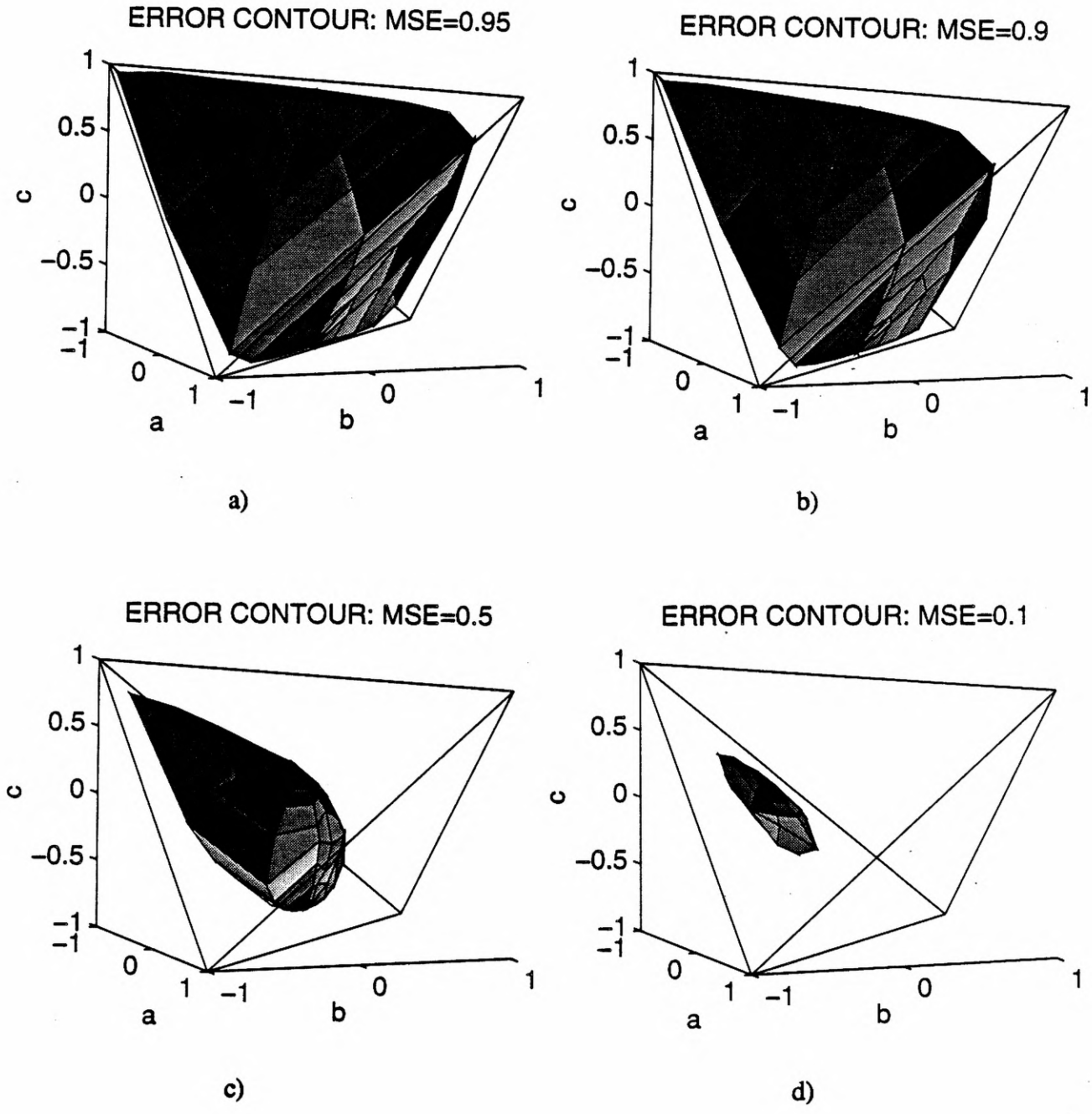


Figure 3.30 Reduced, normalized error contours for the sufficient, first-order 2-D IIR adaptive filter with a) MSE=0.95, b) 0.9, c) 0.5, and d) 0.1 (example 5).

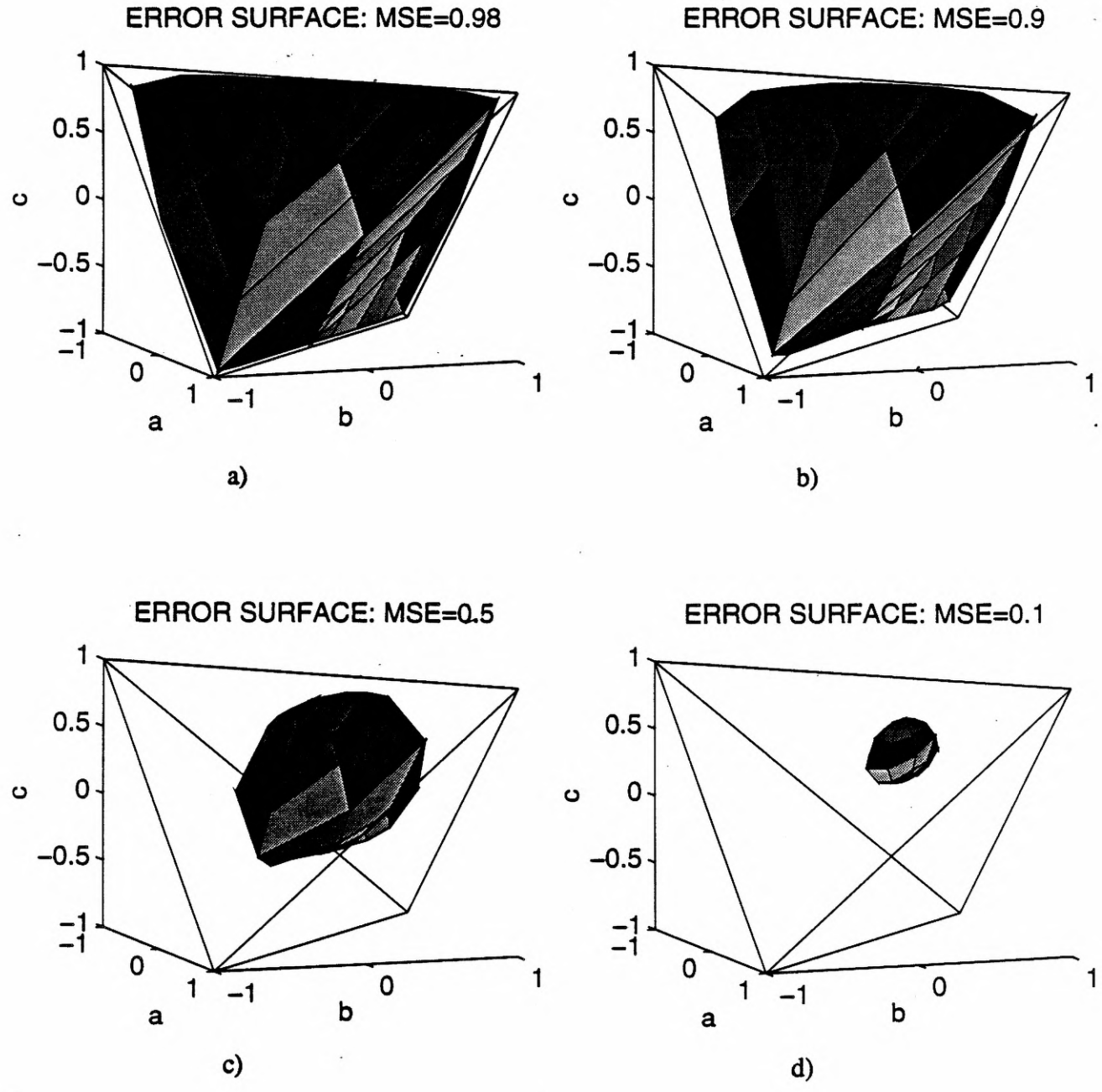


Figure 3.31 Reduced, normalized error contours for the sufficient, first-order 2-D IIR adaptive filter with MSE=0.98, 0.9, 0.5, and 0.1 (example 6).

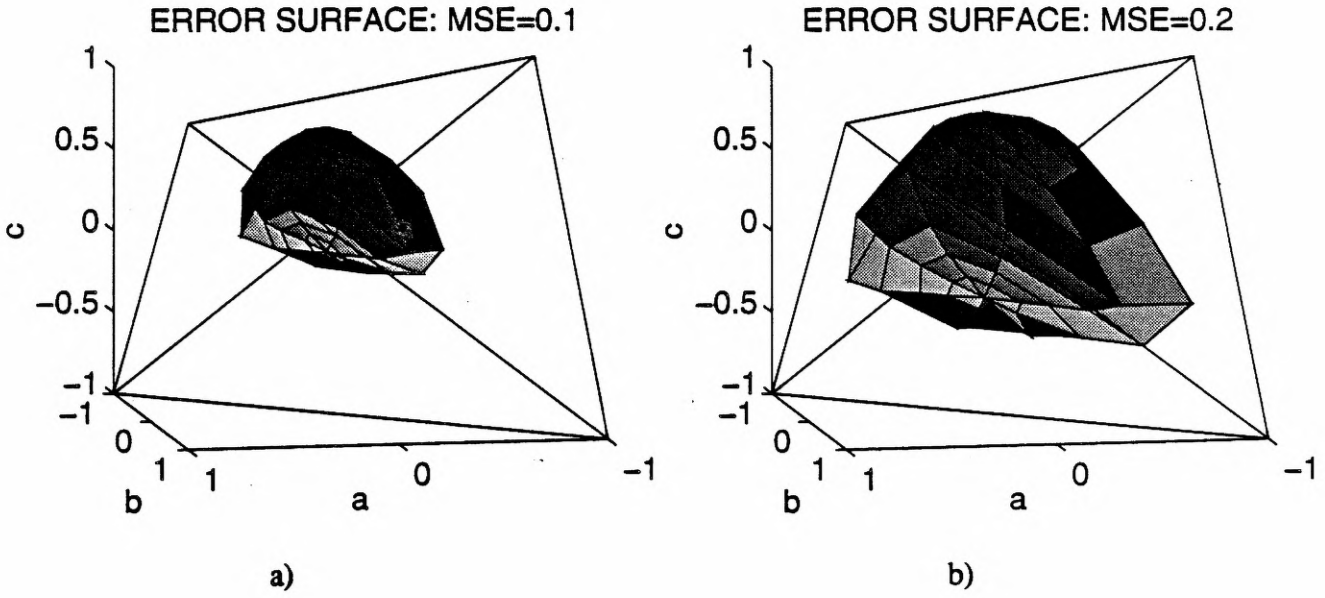


Figure 3.32 Reduced, normalized error contours for the sufficient, first-order 2-D IIR adaptive filter with a) MSE= 0.1, and b) 0.2 (example 7).

CHAPTER 4
IMAGE PROCESSING APPLICATIONS

4.1 Two-Dimensional ADPCM

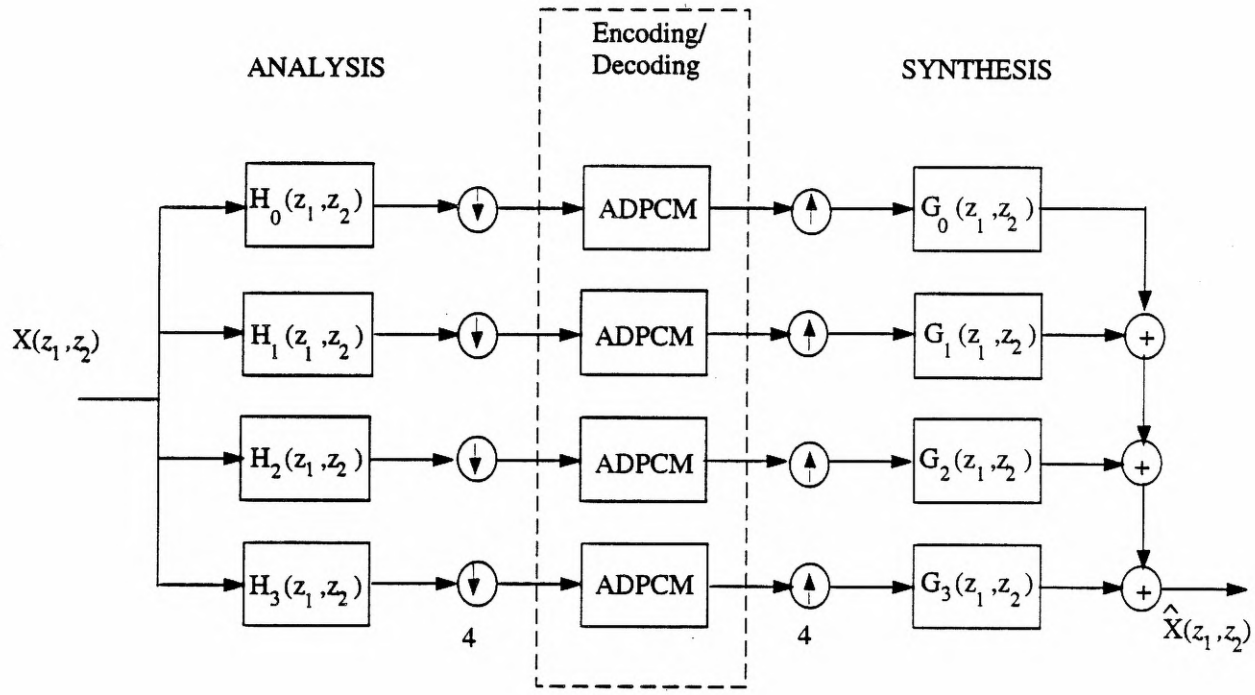
As mentioned earlier, two-dimensional adaptive filtering has as many potential applications as one-dimensional adaptive filtering. Image data compression is important because of the extremely large raw data rates of digital images and video. This is important for transmission as well as storage. Among the methods of compression are predictive coding, transform coding, and subband coding. Adaptive differential pulse code modulation is an example of predictive coding. Transform coding is a block algorithm in which the image is partitioned into blocks, say of size 8x8, and the block is then transformed using any one of a number of transforms [22]. The discrete cosine transform and the discrete Fourier transform are the most common transforms used in this case. Most transforms have the property that much of the signal energy is contained in a relatively few number of the transform coefficients. By assuming a particular zonal mask and choosing an appropriate bit allocation scheme, the image can be coded with fewer bits per pixel. With a fixed mask, perhaps a quarter-plane or triangularly shaped eighth-plane mask, only the nonzero components have to be coded and transmitted. This method has the disadvantage that the reconstructed image has a noticeable blocking effect because each block is processed independently. Therefore, the boundaries of the transform blocks are noticeable. This is usually alleviated with low-pass filtering.

Subband coding is an image compression technique in which different frequency bands of the original signal are isolated with band-pass filters followed by decimation and coding. For the simple 1-D example of two distinct bands, such as low pass and high pass, the signals are decimated by two and coded. When subband filtering images, we can decimate by two in each direction producing four subband signal components, each with 25% as many pixels as the

reference image. From information theory we define the rate distortion function (RDF) for a Gaussian random variable of variance σ^2 and distortion $D=E\{e^2\}$ with respect to some reference as [21]

$$R_D = \max\left[0 \; ; \; \frac{1}{2}\log_2\left(\frac{\sigma^2}{D}\right)\right]$$

If the signal under consideration has significant coloring, then the rate distortion function of the sum of the two distinct subband signals is less than that of the original signal. This happens because we replace the arithmetic mean with the geometric mean, which requires fewer bits to code. An estimate of the original signal is then reconstructed at the receiver using interpolation and synthesis filters as shown in Figure 4.1. The combined system here utilizes both subband filtering and ADPCM. Standard subband coders utilize differential pulse code modulation with a fixed predictor, which is known to be an inferior method of coding compared to ADPCM. The spectral components of the image decimated by 4 are shown in Figure 4.2, and a separable implementation of the subband filtering is shown in Figure 4.3. The bit rate (bpp) can be reduced significantly by exploiting some kind of adaptive bit allocation scheme. Since most images have most of their energy contained within small regions of the two-dimensional frequency domain, different bands can be coded with fewer bits. This usually requires a higher rate of decimation in order to isolate "busy" and "quiet" bands. With many bands requiring no coding, the average bit rate can often be reduced below 1.0 bpp with good visual results. Good performance requires careful design of the analysis and synthesis filters. Parallel and pipelined computing techniques can be exploited, along with separable filtering, the FFT, and frame buffering, to operate the combined subband/ADPCM system in real time. After decimating by a factor of M, the complexity per processor drops by a factor of M with M processors available.



Two-Dimensional Subband/ADPCM
Image Compression

Figure 4.1 Subband coding with 2-D ADPCM.

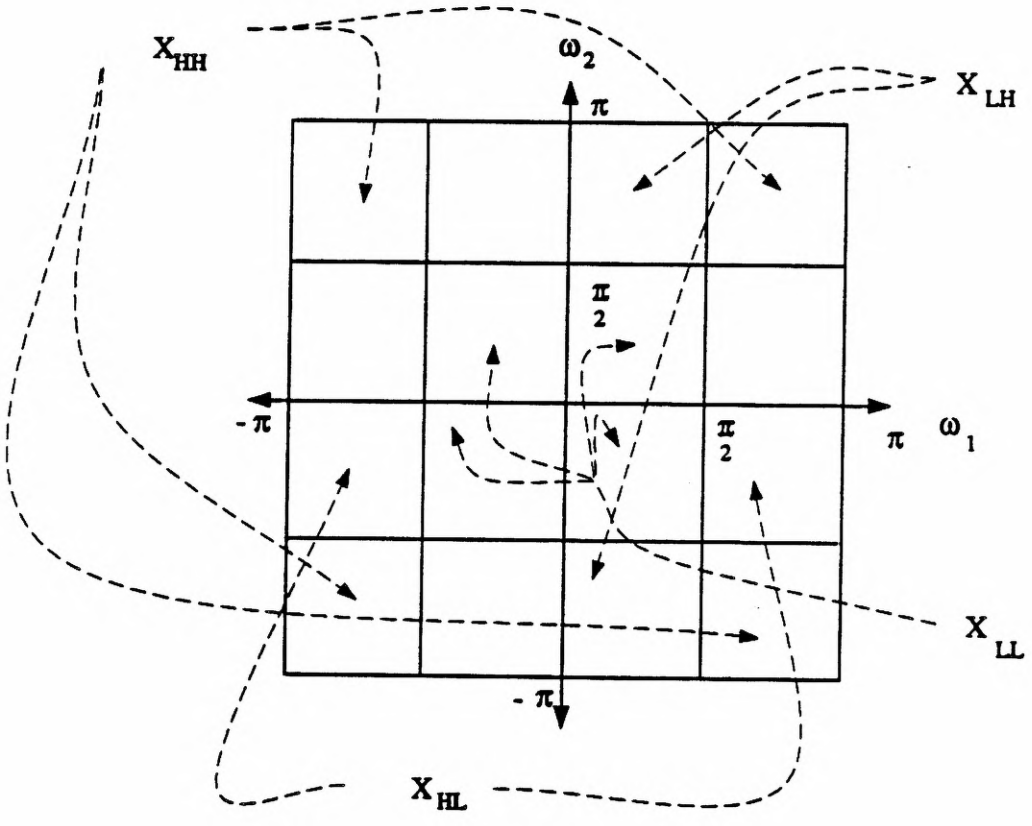


Figure 4.2 Spectral components resulting from decimation by 4.

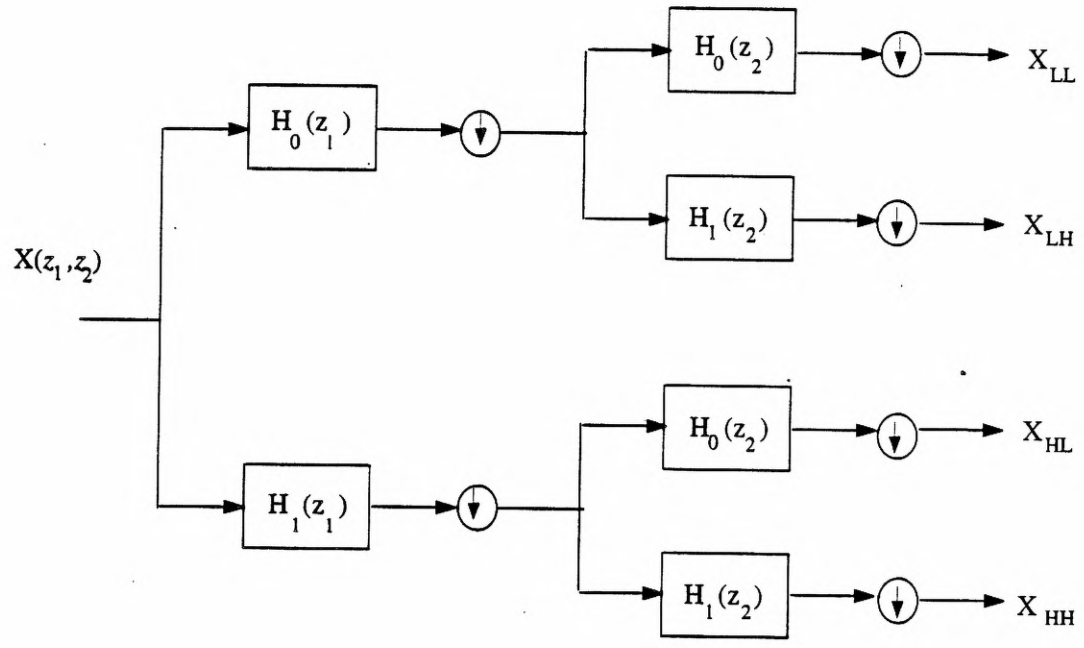


Figure 4.3 Separable implementation of the analysis filter bank for decimation by 4.

Figure 1.1 in Chapter 1 showed the detailed system configuration for 2-D adaptive differential pulse code modulation. Most signal compression techniques exploit the correlation between consecutive samples so that only the difference or error has to be transmitted at each time instance. Since the error signal, obtained by subtracting the predicted signal from the input signal, is expected to have a much smaller variance than the original signal, it can be coded with fewer bits. Telephony standards such as CCITT utilize 1-D ADPCM for speech compression over communication links. Experimental results have shown that the reconstructed image using a two-bit error word length and a fixed quantizer is very nearly visually indistinguishable from the original eight-bit gray level image. We present some experiments illustrating these conclusions here.

Figure 4.4 shows the eight-bit gray level reference image of "Lena" which is to be compressed and then reconstructed using 2-D ADPCM. Two different adaptive algorithms will be used to adjust the coefficients of the recursive prediction filter. First, the IIR gradients are approximated ignoring the recursive nature of the structure to give an FIR LMS update relation. Next, the complete IIR gradients are included giving the familiar IIR coefficient update relation as in Chapter 3. Both cases utilize a first-order denominator and a fixed four-level quantizer. Both methods also use a standard horizontal raster scan. Figure 4.5 shows the reconstructed image using the simplified IIR algorithm giving MSE=31.0. Figure 4.6 shows the reconstruction using the complete IIR update with MSE=26.87. Both cases use the threshold levels (-16, 0, 16) with quantization levels (-30, -6, 6 30), and the convergence factors are experimentally optimized. Upon close examination, errors are noticeable when comparing the reference and reconstructed images, such as slight blurring of sharp edges. Additional performance improvements may be realized by using a predictor with a higher-order numerator along with an adaptive quantizer. Similar results have been obtained with different images. There appears to be little difference between the two algorithms for ADPCM.

Figure 4.4 Reference image #1: eight-bit gray level image of "Lena".

Figure 4.5 Reconstructed image of "Lena" using 2-D ADPCM with the simplified first-order IIR adaptive filter and a two-bit quantizer.

Figure 4.6 Reconstructed image of "Lena" using 2-D ADPCM with the complete first-order IIR adaptive filter and a two-bit quantizer.

4.2 Two-Dimensional Interference Cancellation

Two-dimensional adaptive noise cancellation is shown in Figure 1.2. Just as in the 1-D case, an additive noise signal, $N(n_1,n_2)$, corrupts some desired signal, $s(n_1,n_2)$. The objective is to obtain an estimate of the noise signal and subtract it from the degraded signal in order to recover an estimate of the original signal, $s(n_1,n_2)$. This technique requires that the adaptive filter process another noise signal which is correlated in some way to the original noise source. The correlation filters between the noise source and the adaptive filter input and the measured output signal, $y(n_1,n_2)$, are shown as A^* and B^*, respectively. If the adaptive filter is of sufficient-order to match the correlation filters, then it is possible to exactly cancel the noise from the observed output signal.

The NTSC standard video signal currently used in the United States is an interlaced raster scan with each frame consisting of 525 lines, 480 of which are active. The remaining 45 lines are blanked out for vertical fly back. Since the field frequency is 60 Hz and the frame frequency is 30 Hz, the interlace ratio is 2:1. Other important video signal parameters include the 15.73 kHz horizontal frequency, the 4:3 aspect ratio, and the 4.2 MHz video bandwidth. Typically the RGB color components are transformed into luminance (Y) and chrominance (I, Q) components. These are used to form a composite video signal with the two chrominance components quadrature amplitude modulated on a suppressed color subcarrier, $f_{sc}=455f_{line}/2=3.57954$ MHz. The I and Q signals are band limited to approximately 1.3 MHz and 0.6 MHz, respectively. Because the luminance and chrominance signals are frequency multiplexed with overlapping bands, crosstalk occurs when decoding the signal. This causes high frequency luminance to be decoded as chrominance, and high frequency in-phase chrominance components to be decoded as luminance. Cross-chrominance causes fringes of color around fine detail and sharp diagonal edges. The cross-luminance errors appear on the decoded video screen as a moving dot pattern at sharp color transitions. It is common to assume that Y, I, and Q are mutually uncorrelated.

Shapiro [17] used a 2-D McClellan transformation filter to adaptively cancel luminance-chrominance crosstalk in frequency domain multiplexed video signals. The video chrominance signal when modulated on the color subcarrier overlaps some of the high frequency luminance information, so that when the composite video signal is decoded degradations are apparent. The problem then is one of noise cancellation similar to that in Figure 1.2. Shapiro designed his enhancement system so that the adaptation was done at the transmitter with filter coefficients being transmitted through a side-channel in a portion of the vacant video band width. The final configuration was equivalent to an adaptive spectrum allocation system.

Computer simulations show the ability of 2-D IIR adaptive filters to cancel colored noise when a correlated noise signal is available. A white Gaussian noise signal, $w(n_1,n_2)$, is FIR filtered with $d(n_1,n_2)$ and added to an image, $u(n_1,n_2)$, so the image is degraded as $u'(n_1,n_2)=u(n_1,n_2)+d(n_1,n_2)**w(n_1,n_2)$. Assuming we have access to a correlated signal, such as $v(n_1,n_2)=c(n_1,n_2)**w(n_1,n_2)$, a matched-order IIR adaptive filter is capable of exactly canceling the noise on $u'(n_1,n_2)$ to recover $u(n_1,n_2)$. For the examples that follow, the fixed filters are defined as $C(z_1,z_2)=(1+0.2z_1^{-1})(1+0.2z_2^{-1})$ and $D(z_1,z_2)=0.3(1+z_1^{-1}z_2^{-1})+0.2(z_1^{-1}+z_2^{-1})$, and the white noise signal has variance 20, $E\{w^2(n_1,n_2)\}=\sigma^2=20$, and zero mean. Since the correlation filters are both first-order, the adaptive filter is chosen to be first-order in both the numerator and denominator. Figure 4.7 shows the image of "Lena" with the colored noised added, and Figure 4.8 shows the image after processing it with the adaptive filter. Note that a diagonal indexing scheme is employed starting at the upper left-hand corner of the image, and the convergence factors are $\mu_{num}=\mu_{den}=0.00000005$. The convergence history of the adaptive filter is very clearly visible as the filter is indexed on the diagonal raster pattern. As the processing approaches the main diagonal (lower left to upper right), the variance of the additive noise decreases until it is no longer noticeable. Similar results have been obtained for the other indexing methods examined in Chapter 3.

This example has been presented as an analogy to the chrominance-luminance cross-talk problem discussed earlier. Our simple noise cancellation experiments are conceptually similar to

the crosstalk problem, and we contend that the 2-D adaptive filtering techniques presented in this dissertation can be applied with success either in a receiver-based system or a transmitter-based spectrum allocation system. A detailed examination with real composite video signals and a detailed system design is beyond the scope of this work and is suggested as further research material.

Figure 4.7 Image of "Lena" degraded with colored noise.